Nucleus Pulposus of the Intervertebral Disc: The Effect of Environmental Factors on Metabolic Activity

Lukas M. Jaworski
University of Miami, ljaworski88@gmail.com

Follow this and additional works at: https://scholarlyrepository.miami.edu/oa_dissertations

Recommended Citation
https://scholarlyrepository.miami.edu/oa_dissertations/2428

This Open access is brought to you for free and open access by the Electronic Theses and Dissertations at Scholarly Repository. It has been accepted for inclusion in Open Access Dissertations by an authorized administrator of Scholarly Repository. For more information, please contact repository.library@miami.edu.
NUCLEUS PULPOSUS OF THE INTERVERTEBRAL DISC:
THE EFFECT OF ENVIRONMENTAL FACTORS ON METABOLIC ACTIVITY

by

Lukas M. Jaworski

A DISSERTATION

Submitted to the Faculty of the University of Miami
in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Coral Gables, Florida

December 2019
NUCLEUS PULPOSUS OF THE INTERVERTEBRAL DISC: 
THE EFFECT OF ENVIRONMENTAL FACTORS ON METABOLIC ACTIVITY

Lukas Jaworski

Approved:

Alicia R. Jackson, Ph.D. 
Associate Professor of Biomedical Engineering

Weiyong Gu, Ph.D. 
Professor and Chair of Mechanical and Aerospace Engineering

Chun-Yuh Charles Huang, Ph.D. 
Associate Professor of Biomedical Engineering

Francesco Travascio, Ph.D. 
Assistant Professor of Industrial Engineering

Ramón Montero, Ph.D. 
Assistant Professor in Practice of Biomedical Engineering

Guillermo Prado, Ph.D. 
Dean of the Graduate School
Lower back pain is a major socioeconomic concern in developed nations. Moreover, it is a condition which a majority of people will experience an episode within their lifetime, with a portion becoming chronically afflicted. Degeneration of the intervertebral disc (IVD) is believed to play a critical role in initiating lower back pain. Poor nutrient supplies are implicated as a one of the root causes of this degeneration and much effort has been spent to better elucidate the behavior of IVD cells.

The IVD is the largest avascular structure within the body, relying on diffusion to migrate nutrients in and clear wastes out. Due to this limitation, nutrients concentrations are low within the center of the disc and waste products also accumulate in high concentrations. For most cell niches, this is considered a harsh environment, but completely normal for the cells of the IVD. Understanding how these cells behave under these environmental conditions may elucidate what nutritional and environmental factors lead to aberrant IVD cell behavior. This dissertation will explore the relationship between nutrient conditions and the metabolic adaptations of the IVD cells. Additionally, a custom
bioreactor was constructed in order to study the complex IVD organ as a whole under varied nutritional conditions.

Glucose consumption rate and gene expression of nucleus pulposus (NP) cells was investigated in an agarose gel system over prolonged culture periods with varied oxygen tension and glucose concentration treatments. Glucose consumption rate was found to decrease with increasing oxygen tension and over time but not with changing glucose concentration. Catabolic gene expression increased for all groups over time, with inhibitors of catabolism following suit. Collagen Type I increased in expression with time for the high oxygen tensions while other anabolic genes did not show any consistent trends over the culture period. The optimal reference genes were evaluated and the genes RPL4 and YWHAZ were found to be more stable than the commonly used 18s and GAPDH.

Next the glucose consumption rate was modeled using a more sophisticated method, the Michalis-Menten kinetic model, allowing for increased fidelity in computational modeling and optimizing culture conditions. No differences between oxygen tension were found in the GCR. Gene expression was again analyzed but the limited culture time did not allow for differences to be seen between oxygen tensions. A correlational analysis revealed several targets for further genetic studies.

A custom-built bioreactor was developed and validated to load whole IVD under simulated physiological loading conditions. This reactor was then used to test whole IVDs under free-swelling and dynamic loading conditions. Tissues were compared for their gross composition as well as gene expression differences. No differences were seen between
loaded and free-swelling discs, and both tended toward a catabolic state when examining gene expression. Gross changes in tissue composition were seen briefly in hydroxyproline content before returning to baseline.

The contents of this dissertation greatly enhance the knowledge of IVD NP cell metabolism. Given that these cells play a vital role in disc degeneration and will be critical in regenerative strategies to treat said condition, understanding the behavior of these critical cells is of paramount importance.
# Table of Contents

List of Figures .................................................................................................................. viii  
List of Tables ................................................................................................................... xiv  
Chapter 1 Specific Aims ............................................................................................... 1  
  1.1 Introductory Remarks ............................................................................................ 1  
  1.2 Specific Aims ....................................................................................................... 2  
  1.3 Contents of This Dissertation ............................................................................. 4  
Chapter 2 Background and Significance ..................................................................... 6  
  2.1 Intervertebral Disc Anatomy .............................................................................. 6  
    2.1.1 Gross Anatomy .............................................................................................. 6  
    2.1.2 Biochemical Composition ............................................................................. 7  
  2.2 Nucleus Pulposus Cell Metabolism .................................................................. 8  
    2.2.1 Glucose Metabolism ....................................................................................... 8  
    2.2.2 Anabolism ..................................................................................................... 9  
    2.2.3 Catabolism ................................................................................................... 10  
  2.3 Intervertebral Disc Degeneration ..................................................................... 11  
    2.3.1 Gross Degenerative Changes ....................................................................... 11  
    2.3.2 Biochemical Degenerative Changes ............................................................ 12  
    2.3.3 Metabolic Degenerative Changes ................................................................. 14
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.4 Bioreactors and Whole Organ Culture</td>
<td>15</td>
</tr>
<tr>
<td>2.4.1 Bioreactors</td>
<td>15</td>
</tr>
<tr>
<td>2.4.2 Whole Organ Culture</td>
<td>16</td>
</tr>
<tr>
<td>2.4.3 IVD Whole Organ Culture</td>
<td>16</td>
</tr>
<tr>
<td>2.5 Use of the Porcine Animal Model</td>
<td>17</td>
</tr>
<tr>
<td>2.6 Significance</td>
<td>18</td>
</tr>
<tr>
<td>Chapter 3 Glucose Consumption of Nucleus Pulposus Cells Under Altered Environmental Conditions</td>
<td>21</td>
</tr>
<tr>
<td>3.1 Introductory Remarks</td>
<td>21</td>
</tr>
<tr>
<td>3.2 Methods and Materials</td>
<td>24</td>
</tr>
<tr>
<td>3.2.1 Cell Isolation</td>
<td>24</td>
</tr>
<tr>
<td>3.2.2 Glucose Consumption Rate Measurement Under Varying Oxygen Tension</td>
<td>26</td>
</tr>
<tr>
<td>3.2.3 Glucose Consumption Rate Measurement Under Varying Glucose Concentrations</td>
<td>26</td>
</tr>
<tr>
<td>3.2.4 Statistical Analysis</td>
<td>27</td>
</tr>
<tr>
<td>3.3 Results</td>
<td>28</td>
</tr>
<tr>
<td>3.4 Discussion</td>
<td>30</td>
</tr>
<tr>
<td>Chapter 4 Gene Expression of Nucleus Pulposus Cells Under Altered Environmental Conditions</td>
<td>34</td>
</tr>
<tr>
<td>4.1 Introductory Remarks</td>
<td>34</td>
</tr>
</tbody>
</table>
4.2 Materials and Methods ............................................................................................ 37
  4.2.1 RNA Extraction and RT-PCR ........................................................................... 37
  4.2.2 Validating Reference Genes ............................................................................. 38
  4.2.3 Gene Expression of Matrix Proteins and Notochordal Markers ....................... 38
  4.2.4 Statistical Analysis ........................................................................................... 39

4.3 Results ..................................................................................................................... 39
  4.3.1 Reference Gene Stability .................................................................................. 39
  4.3.2 Gene Expression of Matrix Proteins and Brachyury ........................................ 42

4.4 Discussion ............................................................................................................... 51

Chapter 5 Modeling Nucleus Pulposus Glucose Consumption rate ............................59
  5.1 Introductory Remarks ............................................................................................ 59

  5.2 Materials and Methods ............................................................................................ 61
    5.2.1 Cell Isolation ................................................................................................... 61
    5.2.2 Glucose consumption rate studies .................................................................... 62
    5.2.3 Lactate Assay .................................................................................................. 65
    5.2.4 RNA Extraction and RT-PCR ........................................................................... 65
    5.2.5 Statistical Analysis ........................................................................................... 66

  5.3 Results ..................................................................................................................... 67
  5.4 Discussion ............................................................................................................... 74
Chapter 6 Whole Organ Culture of IVD

6.1 Introductory Remarks

6.2 Materials and Methods

6.2.1 Design of Intervertebral Disc Bioreactor

6.2.2 Preparation of Intervertebral Disc Segments

6.2.3 Viability Testing of Intervertebral Disc Segments

6.2.4 Extracellular Matrix Characterization

6.2.5 Gene Expression

6.2.6 Statistical Analysis

6.3 Results

6.4 Discussion

Chapter 7 Conclusions and Recommendations for Future Work

7.1 Summary and Concluding Remarks

7.1.1 Effects of Oxygen Tension and Glucose Concentration on NP cells

7.1.2 Development of IVD Bioreactor

7.2 Recommendations for Future Work

Works Cited

Appendix A: Technical Drawings

Mechanical Drawings

Electrical Schematics
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appendix B: Source Code</td>
<td>142</td>
</tr>
<tr>
<td>Main Program</td>
<td>142</td>
</tr>
<tr>
<td>Bioreactor Library</td>
<td>147</td>
</tr>
<tr>
<td>Header files</td>
<td>147</td>
</tr>
<tr>
<td>Source</td>
<td>162</td>
</tr>
<tr>
<td>Load Cell Weight Calibration</td>
<td>215</td>
</tr>
<tr>
<td>Python Interface</td>
<td>221</td>
</tr>
<tr>
<td>ADC Library</td>
<td>276</td>
</tr>
<tr>
<td>Header files</td>
<td>276</td>
</tr>
<tr>
<td>Source</td>
<td>289</td>
</tr>
<tr>
<td>I^2C Multiplexer Library</td>
<td>315</td>
</tr>
<tr>
<td>Header files</td>
<td>315</td>
</tr>
<tr>
<td>Source</td>
<td>317</td>
</tr>
</tbody>
</table>
List of Figures

Figure 2-1: a) Model of an intervertebral disc cut transversely, showing the cartilaginous endplate (CEP), annulus fibrosis (AF), and nucleus pulposus (NP). b) Picture of an intervertebral disc cut coronally, the nucleus pulposus is dyed red to better visualize the difference between annulus fibrosis and nucleus pulposus. ................................. 7

Figure 2-2: Diagram of the aggrecan molecule showing the first globular region (G1) which binds to hyaluronic acid and is secured via a link protein (not shown), following the G1 region is a second globular region (G2), which is then followed by the keratin sulfate binding region (KS) and the two chondroitin sulfate binding regions (CS1 and CS2), and finally capped off by a third globular region (G3). The two leftmost images show the change in proteoglycan composition with age, a shortening of the CS chains and a lengthening of the KS chains. The final two images show the typical degradation pattern of the aggrecan molecule. Image taken from Sivan et al. 2014. .............................................................................................. 14

Figure 3-1: Experimental outline for GCR and gene expression experiments used in Chapters 3 and 4........................................................................................................ 25

Figure 3-2: Glucose consumption rate over time given in nmol/million cells/hour. Error bars indicate ± one standard deviation (*p≤0.05, **p≤0.01, and *** p≤0.001). Both graphs show the same data just arranged to more easily label significant differences A) within days or B) within groups. ................................................................. 28

Figure 3-3: Glucose consumption rate over time given in nmol/million cells/hour. Error bars indicate ± one standard deviation (*p≤0.05, **p≤0.01, and *** p≤0.001). Both
graphs show the same data arranged to more easily label and compare significant differences A) within days or B) within groups........................................................ 29

Figure 3-4: Representative LIVE/DEAD of agarose gel disc after ten days of culture.... 30

Figure 4-1: The results of the GeNorm analysis run on all the pigs. A) The average expression stability values $M$ of the remaining genes once the least stable one is removed. B) A pairwise variation analysis between $N\text{F}_n$ and $N\text{F}_{n+1}$ control genes to find the most stable combination of genes, this plot indicates that an average of the six most stable genes is the optimal normalization factor. .............................................. 40

Figure 4-2: The results of the GeNorm analysis run on the pigs in the glucose concentration experiments. A) The average expression stability values $M$ of the remaining genes once the least stable one is removed. B) A pairwise variation analysis between $N\text{F}_n$ and $N\text{F}_{n+1}$ control genes to find the most stable combination of genes, this plot indicates that an average of the seven most stable genes is the optimal normalization factor................................................................................. 41

Figure 4-3: The results of the GeNorm analysis run on the pigs in the oxygen tension experiments. A) The average expression stability values $M$ of the remaining genes once the least stable one is removed. B) A pairwise variation analysis between $N\text{F}_n$ and $N\text{F}_{n+1}$ control genes to find the most stable combination of genes, this plot indicates that an average of the four most stable genes is the optimal normalization factor. ........................................................................................................................................ 41

Figure 4-4: Relative expression of brachyury between a) oxygen tension groups and b) days. Bars between groups indicate significant differences ($p<0.05$). ......................... 44
Figure 4-5: Relative expression of Aggrecan (a, d), Collagen I (b, e), and Collagen II (c, f) between oxygen tension groups (a, b, c) and days (d, e, f). Bars between groups indicate significant differences (p<0.05). ................................................................. 45

Figure 4-6: Relative expression of MMP 1 (a, e), MMP 3 (b, f), MMP 13 (c, g), and MT 1 (d, h) between oxygen tension groups (a, b, c, d) and days (e, f, g, h). Bars between groups indicate significant differences (p<0.05). ..................................................... 46

Figure 4-7: Relative expression of TIMP 1 (a, d), TIMP 2 (b, e), and TIMP 3 (c, f) between oxygen tension groups (a, b, c) and days (d, e, f). Bars between groups indicate significant differences (p<0.05). ................................................................. 46

Figure 4-8: Relative expression of brachyury between a) glucose concentration groups and b) days. ............................................................................................................... 49

Figure 4-9: Relative expression of Aggrecan (a, d), Collagen I (b, e), and Collagen II (c, f) between glucose concentration groups (a, b, c) and days (d, e, f). Bars between groups indicate significant differences (p<0.05). ................................................................. 50

Figure 4-10: Relative expression of MMP 1 (a, e), MMP 3 (b, f), MMP 13 (c, g), and MT 1 (d, h) between glucose concentration groups (a, b, c, d) and days (e, f, g, h). Bars between groups indicate significant differences (p<0.05). ..................................................... 50

Figure 4-11: Relative expression of TIMP 1 (a, d), TIMP 2 (b, e), and TIMP 3 (c, f) between glucose concentration groups (a, b, c) and days (d, e, f). Bars between groups indicate significant differences (p<0.05). ................................................................. 51

Figure 5-1: Experimental overview of Michalis-Menten GCR experiment ..................... 64
Figure 5-2: A) Sample curve-fit of GCR experiment. B) Kinteic plots of GCR rates at various glucose concentrations using calculated $K_m$ and $V_{max}$ values. .......................... 68

Figure 5-3: a) $K_m$, b) $V_{max}$, and c) LPR:GCR of the NP cells for the various oxygen tension groups. No significant differences were found between oxygen tension groups. .... 69

Figure 5-4: A) Brachyury and B) Hypoxia Inducible Factor 1α expression in relation to oxygen tension group. ............................................................................................... 70

Figure 5-5: Expression of A) aggrecan, B) collagen I, and C) collagen II in response to culture in different oxygen tensions ................................................................. 70

Figure 5-6: Expression of A) MMP 1, B) MMP 3, C) MMP 13, and D) MT 1 in response to culture in different oxygen tensions ................................................................. 71

Figure 5-7: Expression of A) TIMP 1, B) TIMP 2, and C) TIMP 3 in response to culture in different oxygen tensions ................................................................. 71

Figure 5-8: Expression of A) GLUT 1, B) GLUT 3, and C) GLUT 9 in response to culture in different oxygen tensions ................................................................. 72

Figure 5-9: Correlational plots of the panel of genes in the study against the transcription factor HIF 1α. All genes had p values less than $10^{-6}$ and the correlation coefficients are listed on the plot of the gene. A) The notochordal marker Brachyury. B) The anabolic genes 1) Collagen I, 2) Collagen II, and 3) Aggrecan. C) The catabolic genes 1) MT 1, 2) MMP 1, 3) MMP 3, and 4) MMP 13. D) The inhibitors of catabolism 1) TIMP 1, 2) TIMP 2, and 3) TIMP 3. E) Glucose transporters 1) GLUT 1, 2) GLUT 3, and 3) GLUT 9 ................................................................. 73
Figure 6-1: Pictographic diagram of the IVD bioreactor. An analog signal from the load cell, made in the whetstone bridge configuration, was read into an ADC on the sensor board and the signal was digitized ($f(ADC)$). This signal was then sent back to the Arduino, which read a real-time clock to adjust the setpoint ($f(set)$). Setpoint data and load data were then fed into a custom PI algorithm used to set output PWM signal ($f(PID)$). Load data, setpoint data, and time were all recorded to and SD card during operation. The PWM signal from the Arduino was sent to the motor controller where it was converted to an analog current signal to drive the motors ($f(DAC)$).

Figure 6-2: Whole IVD bioreactor. Reactor on left does not have the incubation vessel to properly show internal components.

Figure 6-3: Experimental overview of bioreactor experiment.

Figure 6-4: Image of cut IVD segment on calibrated background. Each square has a 1cm length.

Figure 6-5: Loading pattern placed on IVD within the bioreactor. The discs were subject to 8 hours of 0.6MPa ±0.2MPa dynamic loading followed by 16 hours of static compression at 0.2MPa to mimic a diurnal loading pattern.

Figure 6-6: Loading in bioreactor during sample experimental run.

Figure 6-7: Water content of NP (A) and (C) and AF (B) and (D) tissue as percent of total mass in the bioreactor (A) and (B) as well as free-swelling discs (C) and (D).
Figure 6-8: Proteoglycan content of NP (A) and (C) and AF (B) and (D) tissue on a dry mass basis in the bioreactor (A) and (B) as well as free-swelling discs (C) and (D). Proteoglycan content was assessed using the DMMB assay. ................................. 100

Figure 6-9: Hydroxyproline content of NP (A) and (C) and AF (B) and (D) tissue on a dry mass basis in the bioreactor (A) and (B) as well as free-swelling discs (C) and (D). Hydroxyproline content was assessed using the Hydroxyproline assay. ........ 101

Figure 6-10: Gene expression of Brachyury (A) and (B) and HIF-1α (C) and (D) using RT-PCR and the -ΔC_q method in both bioreactor (A) and (C) and free-swelling (B) and (D) discs. .......................................................... 102

Figure 6-11: Gene expression of catabolic genes ADAMTS 4 (A) and (B), ADAMTS 4 (C) and (D), MMP 3 (E) and (F), MMP 13 (G) and (H), using RT-PCR and the -ΔC_q method in both bioreactor (A), (C), (E), and (G) and free-swelling (B), (D), (F), and (H) discs. ................................................................................................................. 104

Figure 6-12: Gene expression of catabolic inhibitor genes TIMP 1 (A) and (B), TIMP 2 (C) and (D), and TIMP 3 (E) and (F) using RT-PCR and the -ΔC_q method in both bioreactor (A), (C), and (E) and free-swelling (B), (D), and (F) discs. ............... 105
List of Tables

Table 4-1: Candidate housekeeping genes................................................................. 38

Table 4-2: Select anabolic, catabolic, inhibitors of catabolism, and Brachyury, a notochordal cell marker. ........................................................................................................ 39

Table 5-1: Panel of genes assessed. List includes anabolic, catabolic, inhibitors of catabolism, glucose transporters, the HIF 1α transcription factor, and the notochordal marker Brachyury. GAPDH is used as the reference gene. .......................... 66

Table 5-2: Results for \( K_m \), \( V_{max} \), and LPR:GCR\(^1\) for oxygen tension groups 2.5\% (n=21), 5\% (n=20), and 21\% (n=20), and overall average for all oxygen levels (n=61). All values are given as mean ±SEM. ........................................................................................................ 69

Table 6-1: Panel of genes assessed. List includes catabolic, inhibitors of catabolism, the HIF 1α transcription factor, and the notochordal marker Brachyury. 18s and RPL4 are used as the reference genes. ........................................................................................................ 97
Chapter 1 Specific Aims

1.1 Introductory Remarks

Lower back pain is a major socio-economic burden, with direct treatment costs being high and an even higher impact due to losses of productivity [1-4]. The etiology of the disease is not clear but degeneration of the intervertebral disc (IVD) is believed to be a major factor in the progression of this condition [5, 6]. The IVD is a complex organ which supports the vertebral bodies, transfers load, and provides the spine with its ability to flex and rotate. The IVD is also the largest avascular structure in the human body, relying on diffusion to move nutrients into the disc and to clear metabolic wastes out [5, 7]. Located in the center of the IVD is the nucleus pulposus (NP), which is a highly hydrated gelatinous proteoglycan matrix with a sparse cellular population [5]. These relatively few cells attempt to maintain the extracellular matrix under the demanding conditions of low nutrient levels and accumulated waste products. In addition, the NP cell population changes phenotypically from birth to adulthood, beginning with notochordal derived cells which are eventually replaced by more chondrocytic cells of unsure origin [6, 8-10]. This change in cell population is usually accompanied by the first degenerative changes seen in the disc [6, 9, 10]. Studying the cells of the NP and how they respond to various environmental cues could help elucidate the progression of disc degeneration and identify potential treatment targets.
The long-term goals of this project are to (1) better understand the pathophysiology involved in the progression of degenerative disc disease; (2) identify the key metabolic factors influencing the disc and disc degeneration; and (3) discover potential targets for treatment and prevention of tissue degeneration. In order to address these goals, the following specific aims will be pursued

1.2 Specific Aims

Specific Aim #1: To determine the effect of glucose concentration and oxygen tension on nucleus pulposus matrix gene expression. Using an agarose gel culture system, the activity of a panel of genes involved in matrix regulation and turnover (MMP 1, MMP 3, MMP 13, MT 1, Collagen Type I, Collagen Type II, Aggrecan, TIMP 1, TIMP 2, and TIMP 3) will be determined, as well as a maker of notochordal stability (T). The gene expression will be evaluated under three oxygen tensions (1%, 5%, and 21%) and three glucose concentrations (1mM, 2.5mM, and 5mM). Moreover, temporal effects of these conditions will be examined with three different exposure times (1, 5 and 10 days) to examine the progression of the changes brought on by adverse conditions. In addition, a two-point glucose consumption rate will be determined to see if there were any gross metabolic changes concurrent with changes in gene expression; a Live/Dead assay will be used to ensure that cell viability is maintained.

Specific Aim #2: To determine the effect of oxygen tension on glucose consumption rate of nucleus pulposus cells. Using an agarose gel culture system, the glucose consumption rate will be evaluated across three oxygen tensions (2.5%, 5%, and 21%). The
Michaelis-Menten reaction kinetics for the glucose consumption rate will be determined to provide a more accurate and dynamic picture of glucose metabolism of the nucleus pulposus cells. In addition, the aforementioned panel of genes will be evaluated to assess changes in matrix metabolism. Additional genes will be added to the analysis to investigate the changes in glucose transporter expression and a hypoxia induced transcription factor (GLUT 1, GLUT 3, GLUT 9, and HIF-1α).

**Specific Aim #3: To determine the cellular response of nucleus pulposus under adverse nutrition and normal loading in whole organ culture.** To this end a whole IVD organ culture system will be developed that maintains consistent nutrient levels, while applying a loading regiment which simulates normal physiological stresses over a prolonged culture time (up to 14 days). Oxygen tension will be controlled to atmospheric (21%) and physiological (5%) levels externally through the use of an incubator with oxygen control capabilities. Discs will also be compared to free-swelling specimens under the same oxygen conditions. Local matrix composition will be evaluated by harvesting the discs and collecting the nucleus pulposus tissue. The collected samples will be evaluated for gene expression of a reduced panel of genes from the previous experiments (MMP 1, MMP 3, MMP 13, TIMP 1, TIMP 2, TIMP 3, T, HIF-1α, and T). In addition, the matrix protein content will be assessed using the hydroxyproline assay to determine total collagen content and dimethylmethylene blue Assay (DMMB assay) for glycosaminoglycan (GAG) quantification. Cell viability will be assessed using a modified Live/Dead assay.
1.3 Contents of This Dissertation

The overall objective of this study was to investigate the metabolism of the NP of the IVD to better understand this unique organ and the pathophysiology of DDD. This understanding can be further used to develop cell based and biologics therapies to better treat the symptoms and underlying conditions of DDD. Background on our current understanding of IVD anatomy, metabolism, and degeneration can be found in Chapter 2. In order to meet the objectives of this study, preliminary metabolic studies were carried out before constructing a whole organ IVD bioreactor to better elucidate the metabolic profile of IVD NP cells.

Initially, Chapter 3 investigates the glucose consumption rate of NP cells under varying glucose and oxygen concentrations to determine how the cells adapt to their \textit{in vitro} environment. This is used to help establish culture condition parameters in later experiments. Results are compared with those in the literature.

The results of Chapter 3 are extended in Chapter 4 with the addition of gene expression data. Candidate reference genes are evaluated for optimal stability and then used to evaluate the expression of key anabolic, catabolic, and inhibitory genes. Additionally, notochordal marker stability is evaluated.

Chapter 5 refines the glucose consumption rate using Michalis-Menten kinetic modeling, and examines whether oxygen tension has any effect on model parameters. The panel of matrix proteins from Chapter 4 are investigated again, with the addition of glucose transporter genes and a transcription factor known to be important in NP cells and
under hypoxic environments. Model results are compared to more traditional approaches in the literature along with changes in gene expression.

In Chapter 6, a custom bioreactor is designed and used to investigate the effect of culture oxygen tension and mechanical loading on whole organ culture of the IVD over a prolonged period. Tissue composition was evaluated for proteoglycan and hydroxyproline content. Once again, the gene expression of key matrix regulating genes are interrogated. The behavior of the NP cells is then compared to others within the literature.

Finally, Chapter 7 summarizes the most impactful results of these data and makes recommendations for future work within the field.
Chapter 2 Background and Significance

2.1 Intervertebral Disc Anatomy

2.1.1 Gross Anatomy

The intervertebral disc (IVD) is a complex organ composed of three main anatomic regions. On the superior and inferior faces of the IVD are the cartilaginous endplates (CEP) composed of hyaline cartilage [11]. They are the anchor points for the vertebra-IVD transition and have collagen fibers that run coronally and sagittally to better serve this function [11]. In between the CEPs lies the nucleus pulposus (NP), a gelatinous matrix comprised of highly hydrated glycosaminoglycans (GAGs) [5, 11]. The GAG composition of the NP allows it to maintain a positive swelling pressure, which is critical to the proper functioning of the IVD [5, 11-14]. Surrounding the NP is a dense fibrous tissue known as the annulus fibrosis (AF), whose fibers are organized 60° off of vertical into concentric lamellae with alternating angle directions [5, 11, 14]. A diagram of these regions can be seen in Figure 2-1. These three major regions work together to provide the spine with motility and help transfer load evenly from one vertebral body to the next by acting like a pressure vessel, where the compressive forces from the CEP region are turned into hydrostatic forces in the NP which is in turn contained by the AF [5, 11]. This allows the IVD to transmit the loads even under bending and torsion.
Figure 2-1: a) Model of an intervertebral disc cut transversely, showing the cartilaginous endplate (CEP), annulus fibrosis (AF), and nucleus pulposus (NP). b) Picture of an intervertebral disc cut coronally, the nucleus pulposus is dyed red to better visualize the difference between annulus fibrosis and nucleus pulposus.

Though the IVD is such an anatomically important organ, it is largely avascular, with capillaries terminating in the vertebral body just above the CEP and in the outer third of the AF [5, 6, 9, 10]. The consequence of this is the IVD relies on diffusion to bring vital nutrients into the disc and to clear metabolic wastes [5, 6, 9, 10, 14-19]. Furthermore, the NP of the IVD relies on diffusion mainly from the CEP route and not the periannular route through the AF, making the NP particularly susceptible to low nutrient conditions and a buildup of metabolic wastes like lactic acid [5, 6, 9, 10, 17-21].

2.1.2 Biochemical Composition

The CEP is composed of dense hyaline cartilage with collagen fibers that run ventrally and laterally into the adjacent bone and disc [11, 14, 22]. With as much as 60% of the dry weight of the NP consisting of proteoglycan, and much of the rest being collagen, the concentration, composition, and structure of these molecules are critical for the correct functioning of the IVD [12-14, 23]. One of the most important proteins in the NP is
aggrecan, which is the main connector of GAGs of the NP: chondroitin sulfate (CS), keratin sulfate (KS), and hyaluronic acid (HA) [24]. Aggrecan binds to hyaluronan monomers present in the disc, which are stabilized by link proteins [11, 13, 14, 23, 24]. The tail ends of aggrecan then bind to CS and KS to create the large aggregates seen within the NP [11, 13, 14, 23, 24]. The high fixed charge density of the NP is maintained by this composition of GAGs which in turn means the NP has high swelling pressure and ensures a highly hydrated matrix [5, 10, 11, 14, 22]. A high swelling pressure is how the healthy disc resists instantaneous loads placed on it by using the incompressibility of water [5, 11-14, 22].

Also present in the NP are some collagen fibers, type II being predominant while type I being mostly absent in healthy discs and trace amounts of a few of the other subtypes are also present [5, 11, 13, 14, 22, 25]. The AF varies from the NP with collagen I being the predominant collagen subtype with type III running parallel to type I fibers [5, 13, 14, 22, 25]. In addition to collagen fibrils, elastin fibers are found between the lamellae of the AF, possibly to restore the structure of the disc following bending or torsion [5, 11, 22]. Though GAGs are found in the AF region of the disc they only make up around 10%-15% of the dry weight composition [11, 13, 14, 24].

2.2 Nucleus Pulposus Cell Metabolism

2.2.1 Glucose Metabolism

Glucose is the main nutrient required to ensure NP cell viability [7, 26, 27]. Though a critical nutrient, the concentration of glucose in the NP is low because the NP relies on diffusion to supply the disc with nutrition [15-17, 20, 21]. Further compounding the energy
balance problem of the NP is the fact that the cells within the NP mostly rely on glycolysis to produce ATP rather than oxidative phosphorylation [7, 27-29]. A consequence of this is that lactic acid also builds up in the NP tissue. Guehring et al. found lactate production rate to glucose consumption rate (LPR:GCR) ratios in the range from 1.7 in notochordal NP cells and 2.1 for mature NP cells [30], while Bibby et al. found the LPR:GCR ratio to be 2.01 for bovine NP cells [7]. Some groups report a positive Pasteur effect (an increase in glycolysis under low oxygen conditions) [28, 31], others report a negative Pasteur effect [7], and still others report no trend at all [32, 33]. The Pasteur effect seen here is mild and glycolysis remains the preferred metabolic route for energy production of the NP cell [7, 28, 30-33].

2.2.2 Anabolism

Maintaining the complex ECM of the NP is a difficult task given the low energy yield of glycolysis. Further complicating matters is that most of the structural proteins are highly glycosylated which further competes for the limited glucose resources of the NP [11, 13, 24]. Since GAG can be up to 60% of the dry weight of the NP, understanding its turnover and production are vital to understanding the functioning of the disc [12, 13, 23, 24]. One of the most important structural proteins within the NP is collagen type II, which is the major type expressed in the NP, with limited levels of types VI, IX, and XI [5, 11, 22, 23, 25, 34]. There is a marked absence of collagen type I in the NP region of healthy discs [5, 11, 22, 23, 25, 34]. This collagen distribution also hints at how the NP functions, favoring the more elastic collagen and arranging in an isotropic mesh to better respond to the hydrostatic pressures generated within the NP.
The other dominant structural protein important for NP structure and function is aggrecan [5, 11, 23, 24, 35]. Aggrecan contains three globular domains, two at the head and one at the tail. The first globular domains’ function is to link the aggrecan protein to hyaluronan, while the function of the remaining two is less clear [24]. In between globular domains 2 and 3 lays a long region of GAG binding sites known as KS, CS1, and CS2. The function of KS is to bind keratin sulfate while CS1 and CS2 both bind chondroitin sulfate, but are distinguished due to a cleavage site between them as well as differing amino acid composition [24].

2.2.3 Catabolism

Catabolic processes are also necessary in the healthy NP to remove damaged matrix and remodel the ECM to best suit the IVD at the current time. This process is important to help clear out older molecules which may have become damaged or undergone undesirable side reactions, such as glycosylation, to which aggrecan is particularly susceptible to [13, 23, 24]. Catabolism in the NP is for the most part handled by a class of enzymes known as matrix metalloproteases (MMPs) and a related family ‘a disintegrin and metalloproteinase with thrombospondin motifs’ (ADAMTS) whose various subtypes target differing ECM components [36].

A study by Mern et al. looked at a host of MMP and ADAMTS protein expression in grade III to grade V degenerated discs and found MMP-3 most strongly expressed with no difference among degenerative grades [36]. In addition, expression of ADAMTS 4 and 5, also known as aggracanase-1 and 2 respectively, had the same magnitude of transcript
copies as MMP 3, though still significantly less [36]. Other detectable MMPs were 1, 2, 7, and 13, though none were in the same magnitude of transcript copies as MMP 3 [36]. In addition, both tissue inhibitor of metalloproteinases (TIMPs) 1 and 2, whose function is to inhibit MMP function [37, 38], was more strongly expressed than MMP 3, showing a strong regulation of this catabolic pathway [36]. TIMPs inactivate MMPs in a 1:1 manner by permanently binding to the protein and blocking the active site [37-42]. Furthermore, MMPs and ADAMTS are translated into their inactive zymogen form and need to be activated before becoming enzymatically active [43-45]. Activated MMP 3 is able to also function as an MMP pro-protein convertase and thus is able to activate other MMPs and ADAMTS molecules [46].

2.3 Intervertebral Disc Degeneration

2.3.1 Gross Degenerative Changes

Intervertebral disc degeneration is a complex process whose etiology is poorly understood. Further complicating matters is that the IVD changes significantly with age and it is difficult to ascribe certain changes to natural aging and others to degeneration [6, 9-11, 14, 16, 22, 24]. One study with human cadaveric subjects over 50 found only a single spine out of fourteen where degeneration had not taken place, and as such considered grade 2 degeneration ‘normal’ for that age group [47]. Though the exact mechanisms and cascade of events that lead to disc degeneration are still unknown there are some commonalities in degenerate discs which allows them to be classified by the level of degeneration. This grading system allows for more consistent descriptions of degeneration and the changes
that the disc experiences, as well as, allows a clinical diagnosis [6, 11, 48]. On a gross analysis basis, the healthy and young IVD will have a clear and distinct transition zone from AF to NP while the higher the grade of degeneration the less distinct this boundary becomes [5, 6, 9-11, 22, 48]. In addition, the fundamental functioning of the NP is altered from a tissue that uses swelling pressure to transfer load, to a tissue that directly resists the load using a fibrous ECM, resulting in a rise of stress concentrations in the IVD [5, 6, 10, 12, 14, 22, 48, 49].

Generally accompanying these changes in the NP are numerous other issues. Changes such as endplate calcification, annular tears, nerve and blood vessel ingrowth, and a loss of disc height and hydration all start appearing as disc degeneration progresses [5, 6, 9-14, 21, 22, 48, 49]. Also, the cell population of the NP changes during the course of aging. Initially the NP cells are large vacuolated cells originating from the notochord, which are individually scattered throughout the ECM. NP cells change to a population of chondrocyte like cells, whose origin is thought to be the endplate but is unconfirmed, which cluster together, have no vacuoles, and do not have notochordal cell markers [6, 9-11, 22, 50, 51]. In addition, the cells in degenerate NP tend to form clusters as opposed to the individual cells of the healthy disc [11, 22, 52-54].

2.3.2 Biochemical Degenerative Changes

As disc degeneration progresses the biochemical content of the IVD changes to adapt to the new conditions present within the disc [8, 10-12, 19, 22-25, 31]. Pearce et al. found that as a whole degenerated discs had higher collagen content, lower water content, and lower
proteoglycan content than their non-degenerate counterparts [47], a trend echoed by several others [9, 13, 14, 22, 25]. Not only is there a change in the gross quantities of ECM molecules, there is also a change in composition in both the collagen types present and a change in the structure of proteoglycan aggregates. Whereas the young and healthy NP has a matrix rich in collagen type II, the mature and degenerate NP sees the relative quantity of this matrix molecule go down while collagen type I increases from undetectable levels to becoming the primary collagen found in the NP[11, 13, 22-24]. In addition, much of the collagen molecules in the degenerated NP tend to be denatured and there is also a higher amount of non-enzymatic crosslinking [13, 22]. With increasing age and degeneration, non-enzymatic glycosylation increases and more of the proteins found in the NP ECM contain these types of changes [13, 22, 24, 50].

Concurrently with changes in the collagen composition of the ECM, there are changes in the aggrecan protein. Though a single protein, this important ECM building block undergoes several important changes with degradation. One of the most readily noticeable changes is the increase in fragmentation of the molecule as parts of the long chain of amino acids are cleaved off, allowing free floating aggregates to enter the ECM, see Figure 2-2 [11, 22-24, 50]. This change is thought to be one of the major causes of proteoglycan and water loss in the IVD [10, 11, 22-24, 50]. In addition to fractioning protein molecules, the composition of the GAGs themselves changes, with KS chains growing longer while the CS chains shrink [24]. All the aforementioned changes lead to the gross changes observed in the disc mentioned earlier. The changes in ECM composition
and crosslinking have a fundamental impact on NP tissue hydration and viscoelastic properties, making the tissue dryer and stiffer [11, 23, 24, 55].

Figure 2-2: Diagram of the aggrecan molecule showing the first globular region (G1) which binds to hyaluronic acid and is secured via a link protein (not shown), following the G1 region is a second globular region (G2), which is then followed by the keratin sulfate binding region (KS) and the two chondroitin sulfate binding regions (CS1 and CS2), and finally capped off by a third globular region (G3). The two leftmost images show the change in proteoglycan composition with age, a shortening of the CS chains and a lengthening of the KS chains. The final two images show the typical degradation pattern of the aggrecan molecule. Image taken from Sivan et al. 2014.

2.3.3 Metabolic Degenerative Changes

Metabolic activity in the IVD is impacted by degenerative state of the disc. One major change is an increase in aggrecan turnover in the IVD; while this may be a self-healing response, a similar change in turnover is not seen in collagen [35]. In addition, the glucose transporters on the surface of NP cells increase with increasing severity degeneration, with GLUT-9 strongly upregulated in intermediately degenerate discs but again almost becoming undetectable at very severe degeneration [29]. This could potentially be to make
use of the limited resources present in the degenerate NP, even though the cellularity slightly decreases, though no direct comparison of glucose consumption between cells of degenerate and healthy discs has been made. Notably, a proliferative assay found no difference in the rate of cell proliferation from grade III to grade V discs [36].

2.4 Bioreactors and Whole Organ Culture

2.4.1 Bioreactors

Bioreactors, in the most general sense, are devices which are used to control culture conditions and enhance a biological response. This broad definition includes applications such as ethanol and antibiotic production, where specific microbe strains are used in a fermentation reaction [56, 57]. In terms of tissue engineering, though, the focus of bioreactors is more in cultivating specific cell subtypes or entire tissues for use in regenerative medicine or to study whole tissue or organ systems under an environment that is more similar to the \textit{in vivo} environment while still being able to precisely control numerous parameters of interest in an \textit{in vitro} setting [58-60]. The specifics of the system parameters are dependent on the goals and the tissues or cells of interest. For example, orthopedic tissues, such as bone, cartilage, tendons, and ligaments, usually have a mechanical loading component in their design, while cardiac tissues are usually electrically or chemically stimulated to produce regular and controlled contractions [58-61]. Currently, the most common techniques in tissue engineering are to either use bioreactors for whole organ culture, where the whole target organ or structure (i.e. a heart valve) is explanted into the bioreactor, or to use them with a seeded scaffold [58-61].
2.4.2 Whole Organ Culture

Whole organ culture is the process which whole organs or tissues are explanted and maintained in an *in vitro* environment. This is distinct from tissue transplantation in that for transplantation the organ or tissue cannot be maintained for long periods of time. One of the main advantages of this technique is the preservation of the native ECM, which can provide numerous chemical and mechanical cues to the surrounding cells [62, 63]. These cues help maintain the native cell phenotype and reduce the risk of dedifferentiation, a problem which plagues cells grown in flasks and even in artificial scaffolds [62-64]. As previously mentioned, bioreactors are employed to maintain the proper nutrient conditions as well as supply other critical physiological cues which the organ or tissue may not supply themselves.

2.4.3 IVD Whole Organ Culture

Though the IVD is avascular, eliminating the need to perfuse the organ through any sort of vasculature, it has some very specific challenges and considerations when designing a whole organ culture system. Due to the high proteoglycan content of the NP, the disc has a tendency to swell and lose proteoglycans if the disc is not confined or the surrounding media osmolarity adjusted [65-67]. In addition to swelling concerns, there are concerns when harvesting the disc that clotting in the bone immediately adjacent to the cartilaginous endplates can prevent nutrient diffusion through the disc’s primary nutrient pathway [66-69]. To navigate around this problem, several approaches have been used. One solution is to use discs where the bone and CEP have been removed; however, this was found to be non-ideal as the AF fiber terminate in the CEP and thus the structural integrity of the disc
was compromised and the NP was able to swell much more than desired [66, 67]. Another solution was to harvest with bone on the CEP and use anticoagulants or power washing, but still some retardation of nutrient flow was noted [70-72]. The final technique investigated was to carefully remove the bone and all calcified CEP but still leave the majority of the CEP intact. This approach was able to maintain disc integrity and minimized nutrient flow problems; however, it is the most difficult technique as the CEP is not a large structure and much of it is fused with the surrounding bone [66, 67, 70].

The next major consideration for IVD organ culture is the external stimuli that needs to be presented to the disc to best simulate the \textit{in vivo} environment. As previously noted, the main function of the IVD is to support and transfer loads between vertebral segments, while maintaining spinal flexibility. NP and AF cells have been found to respond to differential loading, and in fact many groups note markers of degeneration with overloading the IVD [68, 70, 72, 73]. So, for whole organ culture of the IVD, nutrition and loading are the two most important factors to consider when designing a whole organ system.

\textbf{2.5 Use of the Porcine Animal Model}

Due to the limited supply of human tissues, a majority of scientific studies use an animal stand-in. There is an understanding that the results from such a study may not fully reflect what would happen in a human, but much of the preliminary work and understanding is gained from these animal models. Though much information can be obtained from animal research, the model animal must be carefully selected for their utility and relevance to the
questions asked. Throughout this work, the porcine model was chosen. This organism was
chosen because it is a large animal with an IVD similar in size and shape to a human IVD
and has a notochordal cell population within the NP that lasts into adulthood [74-80].
Bovine and ovine models are alternative large animal models, but their NP cell subtype is
predominately chondrocytic in adulthood, while porcine models maintain close to 80%
notochordal subtype within the NP [74-80]. Small animal models, such as the murine and
rabbit models, have abundant notochordal NP cells, but their IVD size is much smaller than
that of a human, and therefore does not share the same nutrient supply deficiencies as larger
animal models [74, 75, 78-80]. The last common animal model is the canine model,
because there are certain breeds which experience similar IVD degeneration to humans
while a majority do not [74, 78, 81, 82].

2.6 Significance
Lower back pain (LBP) is a costly condition that many will experience in their lifetimes
[1-3, 6, 22]. Notochordal NP cells still lack clear consensus in their characterization, but
their disappearance seems to play a vital role in the progression of degenerative disc disease
[9, 81-83]. Further evidence of the importance of notochordal cells in the progression of
disc disease can be seen in the relative rates of symptoms in chondrodystrophoid and
nonchondrodystrophoid dogs, where nonchondrodystrophoid dogs tend to avoid spine
issues and, in particular, degenerated discs [81, 82]. Compounding the problem are
multiple age-related changes in the IVD which make healing more difficult and DDD more
common in a person’s later years [9, 13, 25, 49, 84, 85]. As the population ages and we
continue experience a demographic shift towards a more senior population, the incidence
of LBP will likely increase and the treatment costs for this condition will continue to rise [86, 87].

Current treatment regiments are often very conservative, focusing on rehabilitation exercise and pain management [22, 88, 89]. If the patient does not respond to the conservative treatment, much more drastic measures, such a nucleoectomy and spinal fusion, are undertaken [1, 11, 22, 88]. These surgical procedures have mixed outcomes after surgery and treatments such as spinal fusion accelerate the degeneration of the surrounding discs [1, 11, 88, 89]. The lackluster results of current treatments are spurring the development of prosthesis, biologics, and cellular treatments [11, 36, 75, 88-91]. Much recent work has gone into the development and characterization of cell sources for the potential treatment of DDD [11, 51, 91, 92]. One of the most attractive targets for cell based treatments are notochordal cells of the NP, due to their matrix production and regulatory capacities [83, 91, 93-95]. Notochordal NP cells have been found to alter the metabolism of the more mature chondrocytic NP cells, though they became ineffective once the degeneration progressing in severity [83, 93]. Notochordal NP cells have also been found to influence the migration pattern of chondrocytic NP cells [8, 94]. Furthermore, matrix production by notochordal cells seems to be distinct from chondrocytic cells and may be more appropriate for the intervertebral NP niche [95]. All of these factors taken together make notochordal NP cells an attractive target for future cell therapies. Though potentially useful in treatment, the cells do not proliferate well which limits some of their power, but co-culture systems have been developed that are helping alleviate some of that problem [96, 97]. Even with their drawbacks, understanding the notochordal NP cell and its
behavior under normal and abnormal conditions is of great importance to not only understanding the prognosis and progression of degenerative disc disease but also potentially treating it.
Chapter 3 Glucose Consumption of Nucleus Pulposus Cells Under Altered Environmental Conditions

3.1 Introductory Remarks

Previous studies evaluating degenerate discs show the NP is affected in several major physical properties, such as ECM composition and water content [5, 6, 10, 14, 34, 49, 54, 98-103]. Though the majority of people will show some form of degeneration within their lifetime, the progression of this degeneration is slow and poorly understood [5, 6, 10, 14, 34, 49, 54, 98-103]. Thus, understanding the metabolism of the nucleus pulposus cells will provide valuable insight into the potential causes of disc degeneration, as well as provide valuable information that could be used in tissue engineering efforts. Since the adult disc is avascular, many nutrients are at extremely low concentrations in the center, especially in the case of degeneration [11, 14, 16-18, 20, 21]. Thus, understanding how the cells respond to such conditions can illuminate the degenerative process and help find treatment targets. Additionally, with future cell therapies, understanding the metabolic profile of the cells could optimize their survival and matrix production.

Since one of the main hallmarks of degenerative disc disease is an NP which is stiffer and drier as a consequence of matrix remodeling [5, 6, 16], this in turn changes the diffusive properties of the disc which in general slows the transport of solutes throughout the disc [5, 6, 10, 17-21, 104, 105]. Thus, the glucose consumption rate of the cells will be directly studied to further characterize the metabolism of the cells and whether it is affected by
adverse nutrient conditions present around the cells. This could be especially helpful in future cell therapies to ensure proper nutrition following transplantation for optimal survival and matrix production.

Oxygen is normally a vital nutrient for cells and is used in the chief energy pathway of most cells (i.e. oxidative phosphorylation). In contrast, NP cells mostly rely on glycolysis and produce lactic acid as a result [7, 27, 28, 31]. This use of glycolysis is fairly insensitive to oxygen tension levels [7, 27, 28, 31], and it is uncertain whether it increases or decreases under low oxygen conditions [7, 27, 28, 31, 32].

Glucose is the other main metabolite and the main energy source for most cells in the body. The NP is no different than other tissues in this regard, but diffusive limitations place a strain on the glucose supply available to the cells [20, 21, 28, 106-108]. Glucose concentrations within the center of the disc can be extremely low, especially when the disc is degenerated [19-21, 106]. Further compounding the problem is the NP cells reliance on glycolysis as the main energy pathway [7, 27, 28, 30-32, 109, 110], which provides an extremely low ATP yield relative to glucose consumed using oxidative phosphorylation. Understanding how the cells alter their metabolism in response to the change of this vital nutrient is of critical importance in future cellular therapies, as glucose has been found to be the most critical nutrient in the survival of the NP cell [7, 26-28, 30, 111]. In fact, Horner et al. found that a large variety of offenses can be tolerated by the NP cell as long as they are not coupled with an absence of glucose, a result that seems to be supported by others [7, 26, 27, 30]. The 21% oxygen tension level selected for this experiment reflects typical
culture conditions for many cell lines, while the 5% oxygen tension reflects the oxygen
tension typically found within tissues throughout the body and 1% oxygen tension
represents a low oxygen condition which the NP cells may normally encounter within the
disc [20, 21, 31, 112, 113]. The 5mM glucose concentration represents the concentration
of glucose found on average in the blood of a healthy individual with the lower
concentrations representing the lower concentrations that can be found within the disc [7,
20, 21, 114-116].

In addition, to key knowledge about cellular metabolism, these data will also allow
for more accurate future modeling and will further the limited quantitative data on NP cell
metabolism, especially for notochordal NP cells. These cells seem to be a promising
candidate for future therapeutic use due to their ability to guide the activity of other IVD
cells, even in relatively low quantities [83, 93, 117, 118]. Understanding this basic response
could help in large scale culture efforts as well as determining cell density and nutrient
supplement requirements for cell therapies. Though there are a few studies that measure
the metabolic output of the NP cells, there are still gaps in the knowledge base. Many
studies use bovine or ovine cell sources which exhibit a more chondrocytic or mature
phenotype as compared to the porcine source which maintain a notochordal population into
adulthood [74, 75, 77-80]. This change in cells is still poorly understood, with arguments
about the source of these ‘mature’ NP cells as to whether they migrate from the CEP or
transform from the native population [8, 94, 98].
This experiment observed the GCR response of notochordal NP cells from a large animal model to three different glucose concentrations and three oxygen tensions over a ten-day time frame in an agarose gel model. Using this information, culture conditions for long-term culture can be optimized. In addition, this study may aid in understanding which parameters are more important to regulate during long-term culture of these cells. These specimens were also used to generate gene expression data. The analysis had several components and will be described in Chapter 4.

3.2 Methods and Materials

3.2.1 Cell Isolation

Cells were harvested and seeded into agarose gels according to previously defined protocols [119-122]. Briefly, primary cells were harvested from Yorkshire pigs, which were approximately 4-5 months of age and 90-115kgs in weight, sourced from a local slaughterhouse (Cabrera Farms, Hialeah, FL). The nucleus pulposus tissue was extracted within two hours post-mortem and placed within an enzymatic solution, composed of high glucose DMEM (HG DMEM; Invitrogen Corp., Carlsbad, CA) supplemented with 0.6 mg/mL collagenase (Worthington Biochemical Corp., Lakewood, NJ) and 0.6 mg/mL protease (Sigma Chemical, St. Louis, MO), for 24 hours under continuous agitation in order to digest the tissue. After enzymatic digestion, the solution was filtered through a 70µm cell strainer (BD Biosciences, Bedford, MA). The suspension was then diluted, centrifuged, and resuspended to a concentration of 1x10^7 cells/mL in HG DMEM supplemented with 10% fetal bovine serum (FBS; Atlanta Biologicals, Flowery Branch,
GA) and 1% Antibiotic Antimycotic (AA; Atlanta Biologicals). The suspension was then mixed with 4% agarose (Sigma Chemical), cast into custom molds (discs with d=8 mm) with 100µL per construct, and allowed to solidify, bringing the final constructs to 2% agarose gels containing 5x10^5 cells. An outline of the experiment can be found in Figure 3-1 below.

Figure 3-1: Experimental outline for GCR and gene expression experiments used in Chapters 3 and 4
3.2.2 Glucose Consumption Rate Measurement Under Varying Oxygen Tension

Isolated cells were cultured overnight in HG DMEM with 10% FBS and 1% AA at 37°C, 5% CO₂, 21% O₂. The gels were then divided into three groups depending on oxygen tension level. Groups were cultured in 5mmol DMEM containing 10% FBS and 1% AA at 21% O₂, 5% O₂, or 1% O₂ and 37°C, 5% CO₂; this point marks day 0. All samples were held in separate oxygen-controlled incubators. Media was changed every two days with new media being placed in the incubator an hour prior to media transfer to equilibrate the oxygen level.

At 1, 5, and 10 days, gels were collected, cut into quarters, and placed into individual wells of a 96-well plate. 200µL of 5mmol glucose DMEM without FBS or AA was added and the gels were cultured for 4 hours at 37°C and 5% CO₂ at their respective oxygen tensions. Initial and final glucose concentrations were measured using a custom modified glucometer made from a commercially available blood glucose monitor and testing strips (Accu-Chek Aviva, Roche Diagnostics, Indianapolis, IN) and a sourcemeter (Keithley SourceMeter, Cleveland, OH) to achieve high fidelity glucose measurements. Data were collected using a custom LabVIEW (National Instruments, Austin, TX) script. Viability was assessed using Live/Dead™ staining to ensure viability was kept above 90% Figure 3-4.

3.2.3 Glucose Consumption Rate Measurement Under Varying Glucose Concentrations

Isolated cells were cultured overnight in HG DMEM with 10% FBS and 1% AA at 37°C, 5% CO₂, 21% O₂. The gels were then divided into three groups depending on glucose
concentration level. Groups were cultured in 5mmol, 2.5mmol, or 1mmol DMEM containing 10% FBS and 1% AA at 21% O₂, 37°C, 5% CO₂; this point marks day 0.

At 1, 5, and 10 days, gels were collected, cut into quarters, and placed into individual wells of a 96-well plate. 200μL of 5mmol, 2.5mmol, or 1mmol glucose DMEM without FBS or AA was added and the gels were cultured for 4 hours at 37°C, 5% CO₂, and 21% O₂. Initial and final glucose concentrations were measured as described above. Viability was assessed using Live/Dead™ staining to ensure viability was kept above 90%.

3.2.4 Statistical Analysis

Four pigs (n=4) each were used in both the oxygen tension studies as well as the glucose concentration studies (a total of n=8 for all experiments). Gels from each pig were then split evenly among the experimental groups, with 9 different groups for the oxygen tension experiments (3 timepoints and 3 oxygen tensions) and 9 different groups for the glucose concentration experiments (3 timepoints and 3 oxygen tensions). The oxygen tension experiments had 105 total samples (n=105) and the glucose consumption rate experiments had 91 total samples (n=91) due to differences in cell yields harvested from the pigs. A two-way ANOVA was performed on the glucose consumption rate measurements using base R (v3.6.1) with the independent variables being oxygen tension level and culture time or glucose concentration and culture time, while controlling for individual pig effects; the R models can be found in Equations 3-1 and 3-2. Significance was inferred at p<0.05. The Tukey post-hoc test in base R (v3.6.1) was used to determine significance (p<0.05) between groups.
3.3 Results

The interaction between oxygen tension and culture time was found to be significant (p=0.037). On day 1, no significant differences between groups could be seen. A difference in the GCR emerged on day 5 between the 1% and 21% (p=0.0058) oxygen tension groups. On day 10, the GCR for the 21% oxygen tension group was significantly lower than that of the other two groups (p=0.043 and p=0.0006 for 1% and 5%, respectively), as can be seen in Figure 3-2 below. In addition, the 5% (p=0.024) and 21% (p=0.0004) oxygen tension groups showed a significant decrease in GCR on day 10 as compared to day 1 values. While the 1% group also showed a decrease, it was not found to be significant. The effect of the blocking factor, pig variability, was found to be significant.

Figure 3-2: Glucose consumption rate over time given in nmol/million cells/hour. Error bars indicate ± one standard deviation (*p≤0.05, **p≤0.01, and *** p≤0.001). Both graphs show the same data just arranged to more easily label significant differences A) within days or B) within groups.
The GCR response to differing glucose levels once again showed a decrease similar to that of the oxygen tension experiments at day 10. All GCRs still fell within the ranges of GCRs reported by others, which can be seen in Figure 3-3 [7, 27, 28, 30, 32, 109, 110]. However, for all groups on each day, the differing glucose concentration groups showed no statistically significant differences. Again there was a significant effect noted based on the blocking factor, the source pig.

![Figure 3-3: Glucose consumption rate over time given in nmol/million cells/hour. Error bars indicate ± one standard deviation (*p≤0.05, **p≤0.01, and *** p≤0.001). Both graphs show the same data arranged to more easily label and compare significant differences A) within days or B) within groups.](image)

Viability was maintained above 90%, a representative LIVE/DEAD image can be found in Figure 3-4 below.
3.4 Discussion

As has been seen in previous studies, the NP cell metabolism appears to have a minimal interaction with oxygen tension [7, 27, 28, 31-33]. On day 1, there was no difference detected in the GCR between groups, but by days 5 and 10, the 21% oxygen tension group showed the lowest GCR, which is similar to the results found in the literature [28, 32]. Even with provided much higher oxygen content, the NP cells still rely on glycolysis for the bulk of their energy production, so a falling GCR also indicates a falling use of energy throughout the cell [7, 27, 28, 31]. Other groups report falling protein synthesis rates at atmospheric oxygen tensions, which could explain some of the GCR decline seen at 21% [31, 123]. The 21% oxygen tension is what cells in atmospheric culture
would experience, while the 5% and 1% oxygen tensions more are physiologically relevant conditions [20, 21, 28, 106, 112, 124]. This is indicative that low oxygen tension helps maintain the NP cell phenotype and may be critical for culturing these cells and maintaining their proper metabolic profile to optimize culturing and survival.

The change in GCR over time for the different glucose concentration mimics that of the 21% oxygen tension group, but glucose concentration did not affect the GCR of the cells in any significant way. The effect of the oxygen tension may outweigh any effect related to differences in glucose concentration; an experiment using a more physiological oxygen tension may help clarify this. Though no differences between glucose concentration groups existed, it does point to oxygen tension being a stronger influence on the GCR over timescales of between five and ten days. Each group the GCR dropped from 350-420nmol/10^6 cells/hr to 200-250nmol/10^6 cells/hr, a similar trend to the 21% O_2 group from Chapter 3 which significantly decreased from 250nmol/10^6 cells/hr on day one to 180nmol/10^6 cells/hr on day ten. Though glucose concentration does not seem to regulate GCR on the same scale that oxygen tension does, other groups have found that glucose concentration can play a role in NP cell gene expression and matrix synthesis [109, 125]. The lowest glucose level used of 1mM was higher than the glucose level needed to keep NP cells viable, with the NP cells even being able to survive 24-hour or longer glucose deprivations with no changes in viability[7, 26, 27, 120, 126].

Compared to results in the literature, the GCR on day 1 is higher than the values reported by some, but falls to the lower end of literature values by day 10, especially the
21% oxygen tension group [30, 32, 109, 110]. This change in consumption may be due to an extended adaptation period as the GCR values fall in line with other porcine models. Day 1 values are very similar to those found by Salvatierra et al. who had a short adaptation period [110], while day 10 values are very close to those seen by Guehring et al. and Naqvi et al., whose cells were first cultured for either 7 or 10 days respectively [30, 109]. Bibby et al. and Cisewski et al. found GCRs much lower than those reported by others, but their cell source was bovine caudal discs and adult human IVDs, respectively, a source known for a high proportion of chondrocytic NP cells which are less metabolically active than notochordal NP cells [5, 7, 30, 32, 51, 93-95].

This study is limited by a small sample size (i.e., low number of source pigs), which may introduce biases based on the individual genetic variability of each pig. The model accounts for this variability by using the host pig as a blocking factor. An additional limitation was that the interaction between oxygen tension and glucose concentration were not explored and may be significant in the metabolic behavior of the NP cells, as other combinations of nutrients were found to interact significantly as well indicating a complex NP response [7]. Additionally a culture time of ten days is relatively short compared to the windows needed for culturing the cells for therapeutic uses, thus, an examination of the response of NP cells over even longer timeframes would be desirable [11, 64, 67, 127-129].

Notochordal NP cells also produce IVD matrix differently than chondrocytic NP cells, which tends to be the more desirable matrix production, with the added benefit of
being able to regulate chondrocytic NP cells in co-culture to produce a more appropriate matrix [82, 83, 93, 95, 117, 118, 123]. Additionally, when differentiating stem cells to a more NP lineage, low oxygen tensions drive the cells to a more desirable phenotype [99, 109]; thus, maintaining a low oxygen tension and understanding how the cells respond to this environmental stimulus is critical for future work, especially in regenerative therapies.

GCR can also be used to improve the fidelity of computational models, which are used to simulate conditions within the disc to better understand nutrient distributions and areas where those nutrients are critically low [20, 21, 106, 115, 130].

This study observed the GCR of NP cells over a prolonged period in agarose gels systems under varied oxygen and glucose conditions to better understand the metabolic response of NP cells under common culture and physiologic conditions. These data can be leveraged in better maintaining the NP cells for regenerative strategies and for improving the data for computational models, which can be better used to simulate in vivo conditions.
Chapter 4 Gene Expression of Nucleus Pulposus Cells

Under Altered Environmental Conditions

4.1 Introductory Remarks

Understanding how and when the NP cells activate their remodeling process is of critical importance to finding potential targets for interventions and treatments. Though this process is studied quite extensively, the degenerative cascade and exact mechanism of degeneration still elude description [5, 6, 9, 22, 79, 80, 131]. One of the main complications in studying this degenerative process is its slow onset and progression, being largely asymptomatic for many with a few experiencing chronic debilitating pain [2, 5, 6, 22]. One of the major changes seen in the matrix of the NP from healthy to degenerative states is the change from collagen Type II as the primary collagen subtype to collagen Type I [25, 68, 84, 131, 132]. In addition to this transition, many note increased expression from several MMPs in regions associated with the degeneration [68, 84, 132]. Though MMPs can be a powerful catabolic force in the disc, they are regulated with TIMPs which inhibit the active form of the MMP in a 1:1 manner [37, 41, 42]. Tracking a panel of several of the main enzymes in NP tissue remodeling, some insight into the beginning stages of the degenerative cascade may be elucidated and targets for biologic and gene therapies can be explored [50, 90, 133]. In fact, one study has already explored TIMP 1 as a gene therapy in rabbits and found moderate success in slowing the progression of simulated DDD with this therapy [133].
There are many studies on the gene expression of NP cells from many different animals and humans [42, 52, 68, 69, 101, 125, 131, 134-146]. In these studies, when performing real time RT-PCR, two housekeeping genes are routinely chosen: 18s [42, 68, 69, 101, 134, 135, 145] (a ribosome transcription factor [147]) and GAPDH [141-144, 146] (an enzyme involved in glycolysis [148]). For accurate real time RT-PCR results, a good housekeeping gene needs to be chosen, specifically one which is stable and does not alter its expression with varying experimental conditions [140, 149-152]. Several methods have been developed to evaluate the stability of suitable candidate genes [150, 152-154]. Additionally, more than one candidate gene can be selected and averaged together to create a ‘pseudogene’ that can be used as a reference [150, 152]. Thus, while the selection of suitable reference gene(s) is critical in providing more accurate results, to our knowledge, no previous study has performed reference gene selection experiments in porcine NP cells, and very few reference gene selection experiments have been done for other animals [140, 149, 151]. Due to similarities, candidate reference genes were selected from a list of reference genes used in chondrocyte studies [140]. In order to find the most robust set of reference genes, data from both the oxygen tension and glucose concentration were analyzed, together and then separately, to see what effect differing experimental conditions had on reference gene stability.

As glucose is the main energy source for the intervertebral disc, understanding its influence on the behavior of IVD cells is a critical piece of information going forward. Since glucose concentration can reach extremely low levels within the disc [11, 14, 18, 20, 21], interrogating how the NP cells respond with matrix remodeling to these conditions
could shed some light on the still poorly understood IVD degeneration process. In addition, glucose has been found to be the most vital nutrient in maintaining NP cell viability over time [7, 26, 71, 155]. FBS also maintained cell viability and promoted protein production and cell proliferation, but FBS also contains glucose along with numerous other proteins and signaling molecules and in general is a poorly characterized media additive with an unknown method of action in these cases [156].

While physiologically the oxygen tension of the NP region of the IVD falls to extremely low levels, cell culture is commonly carried out at atmospheric conditions [20, 21, 106, 112, 124]. The need for a low oxygen tension environment increases complexity of both experiments and potential production of therapeutic cells, requiring specialized equipment and procedures to monitor and adjust oxygen levels. This is, however, necessary for NP cells, as our earlier study on the GCR (Chapter 3) showed oxygen tension effects cellular metabolism; other studies have shown similar results, as well as changes in protein synthesis, lactate production, and extracellular matrix remodeling [7, 31, 109, 123, 124, 156, 157]. Thus, understanding how the gene expression of key matrix remodeling proteins as well as monitoring the stability of the notochordal cell lineage could provide key insights for targeted therapies, in terms of gene therapies, cell therapies, and biologics [50, 100, 137, 158].

This study investigates the notochordal NP cell response of a large animal model to altered glucose and oxygen conditions over a 10-day culture period. Optimal reference genes are evaluated for improved gene expression results. Changes in catabolic and anabolic activity
are noted in this *in vitro* system over a prolonged culture time. The results can be used to better select reference genes in the future for more accurate gene expression studies. Additionally, the results can be used to optimize notochordal cell culture strategies, improving cell retention and desired ECM production. Trends in anabolic and catabolic activity can be used to find potential treatment targets for biologic therapies as well as potential triggers of the degenerative cascade of DDD.

### 4.2 Materials and Methods

The data presented here uses the same cells used in the GCR experiments from Chapter 3. All of the groups are the same and experimental treatment conditions for all gels can be found in Chapter 3 Subsection 2.1-2.4. Also, an overview of the experiment can be found in Figure 3-1. Cells were then used for validating reference genes and examining gene expression of relevant matrix proteins and notochordal markers, described below.

#### 4.2.1 RNA Extraction and RT-PCR

Total RNA was extracted using TRIzol (Tri-Reagent, Molecular Research Center, Cincinnati, OH) protocol. Total RNA was quantified using the Qubit RNA BR assay kit (Life Technologies, Carlsbad, CA) and reverse transcribed to cDNA using the High Capacity cDNA reverse transcription kit (Applied Biosystems, Foster, CA), according to manufacturers’ specifications. Gene expression was quantified through the use of real time-PCR (One step Plus, Applied Biosystems).
4.2.2 Validating Reference Genes

A list of the reference genes studied can be found in Table 1. Genes were analyzed using the NormqPCR package [159] (1.7.1) in R [160] (3.3.2) to apply both the GeNorm algorithm to the whole data set and to each sub portion in order to rank the most stable genes, and to see if there are any variations based on experimental conditions.

Table 4-1: Candidate housekeeping genes.

<table>
<thead>
<tr>
<th>Gene</th>
<th>Forward Sequence</th>
<th>Reverse Sequence</th>
<th>Target Gene</th>
</tr>
</thead>
<tbody>
<tr>
<td>18s</td>
<td>CGGCTACCACATCCAAG GA</td>
<td>AGCTGGAATTACCG CGGCT</td>
<td>NR_046261.1</td>
</tr>
<tr>
<td>GAPDH</td>
<td>GTTTG TGATGGCGTG TAACC</td>
<td>AGCTTGACGAAGTGCT CGTT</td>
<td>NM_001206359.1</td>
</tr>
<tr>
<td>ACTB</td>
<td>GTTCGAGACCTTCAACAC GC</td>
<td>GCCTAGAAGCA TTTGCGGTG</td>
<td>DQ845171</td>
</tr>
<tr>
<td>RPL4</td>
<td>ATACAGACCTTGAAGAATCTTGA</td>
<td>AATCTTTCTTGCGTG TGGCG</td>
<td>DQ845176</td>
</tr>
<tr>
<td>TBP</td>
<td>ACTGTGCTGCTATTTTGGGCA</td>
<td>TGAAAACCGCGGAATGTGTCTG</td>
<td>DQ845178</td>
</tr>
<tr>
<td>HPRT1</td>
<td>TATGGACAGGACTGAACGGC</td>
<td>TCCAGCAGGTCA GAAAGA</td>
<td>DQ845175</td>
</tr>
<tr>
<td>HMBS</td>
<td>CGAGAGTGCCCCTATGTAGC</td>
<td>GTGTTTGATGGGTT CCCGGA</td>
<td>DQ845174</td>
</tr>
<tr>
<td>YWHAZ</td>
<td>AAACAGCAGATGGCTCGAGAA</td>
<td>CTGCTTGAGCATTTGGGG</td>
<td>DQ845179</td>
</tr>
</tbody>
</table>

4.2.3 Gene Expression of Matrix Proteins and Notochordal Markers

A list of the genes studied can be found in Table 4-2. The values were transformed to ΔCq values using the NormqPCR package [159] (1.7.1) in R [160] (3.3.2). Significant differences between groups and days were analyzed using a two-way ANOVA in base R controlling for individual pig variance [160].
Table 4-2 Select anabolic, catabolic, inhibitors of catabolism, and Brachyury, a notochordal cell marker.

<table>
<thead>
<tr>
<th>Gene</th>
<th>Forward Sequence</th>
<th>Reverse Sequence</th>
<th>Target Gene</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMP 1</td>
<td>ATTTGTGGGAGCCCCAGA GT</td>
<td>CCGAGCTTTGAACCCCTTT</td>
<td>NM_213857.1</td>
</tr>
<tr>
<td>TIMP 2</td>
<td>CTAGGCAACCCCCATCAAAC</td>
<td>AGAGGAGGCCCCTGAGA TATA</td>
<td>NM_001145985.1</td>
</tr>
<tr>
<td>TIMP 3</td>
<td>CCTGCCCTGCTTTGTGACCT</td>
<td>CGGATGACCGCTGCTGTGTTT</td>
<td>XM_003126073.4</td>
</tr>
<tr>
<td>MT 1</td>
<td>CTGGACTGTCGGGAATGAGG</td>
<td>AGGGTCAATGGAGCTGAC TCA</td>
<td>NM_214239.1</td>
</tr>
<tr>
<td>MMP 1</td>
<td>GACCTGGAGGAACCTCTT GCT</td>
<td>GCCTGGATGCCCACTCAT GTC</td>
<td>NM_001166229.1</td>
</tr>
<tr>
<td>MMP 3</td>
<td>TCCTGATGTTTGTTACCTCAGCAC</td>
<td>TTGACAATCTGAGATGAGGC TATT</td>
<td>NM_001166308.1</td>
</tr>
<tr>
<td>MMP 13</td>
<td>CATGAGTTGGCCACCCCTT</td>
<td>GTGGCCTTTGCGAACGTGAGG</td>
<td>XM_003129808.3</td>
</tr>
<tr>
<td>Aggrecan</td>
<td>CAGTGAAGACTTGTGGACATC</td>
<td>GTGATGACGGGAGGCACCC</td>
<td>NM_001164652.1</td>
</tr>
<tr>
<td>Collagen I</td>
<td>TGAGCTGATCTGGCCACGAAA</td>
<td>GCACCAATCAATCTCAGACGAC</td>
<td>XM_005668927.1</td>
</tr>
<tr>
<td>Collagen II</td>
<td>CACGGATATCCTCACCAGG</td>
<td>TCGGGCCTTTTCTCCACCAC</td>
<td>XM_001925959.5</td>
</tr>
</tbody>
</table>

4.2.4 Statistical Analysis

The gene expression data were analyzed using a two-way ANOVA with a blocking factor on the pig to control for individual variation in expression between pigs. The model in this experiment was the same as in Chapter 3.

4.3 Results

4.3.1 Reference Gene Stability

Though GAPDH and 18s are commonly used as the housekeeping genes of reference in many RT-PCR experiments, they were not found to be the most stable reference genes from the panel assessed, falling somewhere in the middle, see Figure 4-1. RPL4, a ribosomal protein [147], and YWHAZ, a hub protein for many signal transduction pathways [161], were the overall most stable genes assessed in these two series of experiments. A slight difference was observed in the ranking of the most stable genes when comparing the effect of oxygen tension versus glucose concentration, see Figure 4-2 and Figure 4-3. The gene stability ranking for the series of experiments were oxygen tension
was varied had many of the gene stability rankings similar to the rankings seen when considering the dataset as a whole. The experiment where glucose concentration was the independent variable showed a major deviation from the ranking trend of the overall dataset. In the glucose concentration experiments, HMBS, a deaminase involved in heme production [162], was the most stable gene analyzed. YWHAZ and RPL4 were still at the front of the stability rankings, at 2 and 3 respectively. Other deviations can be seen in

**Figure 4-2**

![Figure 4-2](image)

**Figure 4-1:** The results of the GeNorm analysis run on all the pigs. A) The average expression stability values M of the remaining genes once the least stable one is removed. B) A pairwise variation analysis between NF\_n and NF\_n+1 control genes to find the most stable combination of genes, this plot indicates that an average of the six most stable genes is the optimal normalization factor.
Figure 4-2: The results of the GeNorm analysis run on the pigs in the glucose concentration experiments. A) The average expression stability values $M$ of the remaining genes once the least stable one is removed. B) A pairwise variation analysis between $N_{F_n}$ and $N_{F_{n+1}}$ control genes to find the most stable combination of genes, this plot indicates that an average of the seven most stable genes is the optimal normalization factor.

Figure 4-3: The results of the GeNorm analysis run on the pigs in the oxygen tension experiments. A) The average expression stability values $M$ of the remaining genes once the least stable one is removed. B) A pairwise variation analysis between $N_{F_n}$ and $N_{F_{n+1}}$ control genes to find the most stable combination of genes, this plot indicates that an average of the four most stable genes is the optimal normalization factor.
4.3.2 Gene Expression of Matrix Proteins and Brachyury

Results for the expression of notochordal marker brachyury (T) are shown in Figure 4-4. Only the 21% oxygen tension group showed a significant decline in T expression on day 10, compared to day 1 (p=0.012) and day 5 (p=0.0071). The 5% group trended down over time while the 1% oxygen tension group trended up, but these trends were not found to be statistically significant. Additionally, the 1% oxygen tension group had a significantly higher T expression than both the 21% (p<<0.001) and 5% (p=0.0032) groups on day 10. There was however a significant interaction between time and oxygen tension (p=0.0008).

Relative gene expression for the 10 genes investigated is shown in Figures 4-5 to 4-7. There were no statistically significant differences noted in aggrecan expression, neither between oxygen tension levels nor over time. Collagen I expression showed a significant increase in expression between days 1 and 5 (p=0.032) for the 5% oxygen tension group, with no other comparisons being significant, though the interaction term was significant (p=0.045). Additionally, collagen Type I was the only gene which had to have their Cq numbers set to 40 (i.e., undetectable expression). This was the case for the majority of samples, especially those analyzed on day 1, which showed no expression during RT-PCR. The 1% oxygen tension group showed a significant increase in collagen II expression between days 1 and 10 (p=0.010) and had significantly higher expression of collagen II as compared to the 21% oxygen tension group on day 10 (p=0.0089). The interaction term was also found to be significant (p=0.0004).
Membrane type 1 matrix metalloproteinase (MT 1 or MMP 14) was the only catabolic gene which showed no significant differences in expression over time or between oxygen tension groups, but did show a significant interaction (\( p=0.0096 \)). The other catabolic genes had upward trends over time for all oxygen tension groups. Significant increases were found between days 1 and 10 for all oxygen tension groups for MMP 1 (\( p=0.0073, 0.0004, \) and 0.0013 for 1%, 5%, and 21% respectively) and additionally between days 5 and 10 for the 21% oxygen tension group (\( p=0.0078 \)). MMP 3 expression was higher on day 10 compared to days 1 (\( p<0.001 \)) and 5 (\( p=0.0006 \)) for the 5% oxygen tension group. On day 1, the 21% oxygen tension group had significantly higher expression than the 5% oxygen tension group (\( p=0.024 \)). Additionally, the interaction term was found to be significant (\( p=0.0232 \)). For MMP 13, all oxygen tension groups showed increases in expression between days 1 and 10 (\( p=0.047, 0.0049, \) and 0.0016 for 1%, 5%, and 21% respectively), with the 21% oxygen tension group showing an increase between days 1 and 5 as well (\( p=0.049 \)).

Though the TIMPs showed similar trends compared to the MMPs, the increases were milder. TIMP 2 showing a significant increase in the 1% oxygen tension group between days 1 and 10 (\( p=0.0034 \)). TIMP 3 significantly increased in the 21% oxygen tension group between days 1 and 5 (\( p=0.040 \)). Significant differences between individual groups and reference levels are noted in the figures.
Differences between pigs were significant in the following genes: aggrecan, collagen Type II, MMP 1, MMP 3, MMP 13, T, TIMP 1, and TMP 3 for the oxygen tension groups.

Figure 4-4: Relative expression of brachyury between a) oxygen tension groups and b) days. Bars between groups indicate significant differences (p<0.05).
Figure 4-5: Relative expression of Aggrecan (a, d), Collagen I (b, e), and Collagen II (c, f) between oxygen tension groups (a, b, c) and days (d, e, f). Bars between groups indicate significant differences (p<0.05).
Figure 4-6: Relative expression of MMP 1 (a, e), MMP 3 (b, f), MMP 13 (c, g), and MT 1 (d, h) between oxygen tension groups (a, b, c, d) and days (e, f, g, h). Bars between groups indicate significant differences (p<0.05).

Figure 4-7: Relative expression of TIMP 1 (a, d), TIMP 2 (b, e), and TIMP 3 (c, f) between oxygen tension groups (a, b, c) and days (d, e, f). Bars between groups indicate significant differences (p<0.05).
For the glucose concentration experiments, no significant differences were noted in T expression over time or between groups, seen below in Figure 4-8. Relative gene expression for the remaining 10 genes investigated is shown in Figures 4-9 to 4-11. The anabolic genes, seen in Figure 4-9, had no statistically significant differences noted in aggrecan expression, neither between glucose concentration groups nor over time. Collagen I expression showed a significant increase in expression between days 1 and 10 (p=0.032) for all three glucose concentrations (p<<0.001, p=0.010, and p<0.001 for 1mM, 2.5mM, and 5mM, respectively). The 1mM group also had a significant difference between days 1 and 5 (p=0.0044). All groups were showing an upward trend for collagen Type I expression. Collagen Type II initially decreased in expression on day 5 before recovering on day 10 for all groups. The differences between day 1 and 5 were found to be significant in the 1mM and 2.5mM groups (p<0.001 and p<<0.001, respectively). For the 5mM group, collagen Type II significantly increased between days 5 and 10 (p=0.014). Additionally, collagen Type I was again the only gene with undetectable expression (i.e., Cq numbers set to 40), similar to the oxygen tension experiments. This was the case for the majority of samples, especially those analyzed on day 1, which showed no expression during RT-PCR.

All catabolic genes had some increases in gene expression over time, as seen in Figure 4-10. MMP 1 had significant increases between days 1 and 10 for all three glucose groups (p<0.001, p=0.0036, and p=0.0015 for 1mM, 2.5mM, and 5mM, respectively). MMP 3 also significantly increased between days 1 and 10 for all three groups (p<<0.001, p<<0.001, and p<0.01 for 1mM, 2.5mM, and 5mM, respectively). The differences between days 5 and 10 were also found to be significant for the 1mM and 2.5mM groups (p<<0.001.
for both). The trend continues for MMP 13 with day 10 being significantly higher than both days 1 (p<<0.001, p<<0.001, and p<0.01 for 1mM, 2.5mM, and 5mM, respectively) and 5 (p=0.0040, p<<0.001, and p<0.01 for 1mM, 2.5mM, and 5mM, respectively) for all three groups. For MT 1 there was a significant difference between days 5 and 10 for the 1mM group (p=0.036).

The TIMPs followed the trends of the catabolic genes and all TIMPs increased over time for all glucose groups. There was also a significant increase in expression between the 5mM and 2.5mM glucose groups for TIMP 3 on day 5 (p=0.043) and there was a significant interaction found between culture time and glucose concentration (p=0.0016). Gene expression of TIMP 3 significantly increased between days 1 and 10 (p<<0.001, p<<0.001, and p<0.01 for 1mM, 2.5mM, and 5mM, respectively) and days 5 and 10 were also found to be significantly higher for all groups (p<<0.001, p<<0.001, and p=0.014 for 1mM, 2.5mM, and 5mM respectively). For TIMP 1, all glucose groups showed significant increases between days 1 and 10 (p<<0.001, p<<0.001, and p<0.01 for 1mM, 2.5mM, and 5mM, respectively) and days 5 and 10 were also found to be significantly higher for the 1mM and 2.5mM groups (p<<0.001 for both). For TIMP 2, again all groups significantly increased between days 1 and 10 (p<<0.001 for all). Days 5 and 10 were also found to be significantly higher for the 1mM and 2.5mM groups (p=0.025 and p=0.0052 respectively).

Differences between pigs were significant in the following genes: aggrecan, collagen Type II, MMP 1, MMP 3, MMP 13, MT 1, T, TIMP 1, TIMP 2, and TMP 3 for the oxygen tension groups.
Figure 4-8: Relative expression of brachyury between a) glucose concentration groups and b) days.
Figure 4-9: Relative expression of Aggrecan (a, d), Collagen I (b, e), and Collagen II (c, f) between glucose concentration groups (a, b, c) and days (d, e, f). Bars between groups indicate significant differences (p<0.05).

Figure 4-10: Relative expression of MMP 1 (a, e), MMP 3 (b, f), MMP 13 (c, g), and MT 1 (d, h) between glucose concentration groups (a, b, c, d) and days (e, f, g, h). Bars between groups indicate significant differences (p<0.05).
4.4 Discussion

This study investigated the impact that oxygen tension and glucose concentration have on the behavior of NP cells over prolonged culture. Specifically, GCR was investigated in Chapter 3 while gene expression was examined here. These insights will be useful to later regenerative efforts in optimizing culture conditions to best maintain phenotype and viability. To our knowledge no examination of the optimal housekeeping genes for porcine NP cells exists in the literature. Thus, the first task undertaken is a quantification of optimal reference genes using an established ranking algorithm [150]. These genes were then used
as the reference for the remaining gene expression studies. While several in vivo studies have examined the IVD and more specifically NP properties over prolonged timeframes, they have been carried out in small animal models with few large animal models in the literature [76, 141, 163, 164]. The porcine model was chosen because it is a large animal model that has discs which experience the same diffusion concerns as humans, while still using a notochordal cell population into adulthood [30, 77, 79, 80, 83]. While small animals are useful, they do not fully reflect the conditions present in a human disc, being too small to have the same concerns about diffusion limits. The notochordal cell is of particular interest due to its relative absence in degenerate discs, being replaced by the chondrocytic phenotype; how this change happens is not firmly established [4, 30, 76, 77, 98, 164, 165]. Because the NP cell has low proliferative capacity, understanding how these cells respond in culture will help future regenerative strategies.

Because the housekeeping gene is the reference point against which gene expression is measured, and PCR amplification is on a logarithmic scale, small perturbations can lead to large errors in the final results. As such, a pairwise variation analysis was run to find the most stable combination of genes. For both sets of data the pairwise variation never dropped below the 0.15 threshold, a not uncommon result [140, 151], and required six of eight candidate genes to achieve maximum stability. Using the whole dataset, the maximum stability achieved was only 0.47. The glucose concentration subset was able to reach only 0.42, requiring seven of the eight housekeeping genes, and the oxygen tension subset managed to reach 0.31, requiring only four genes for maximum stability.
Though 18s and GAPDH are commonly used housekeeping genes of reference in gene expression studies of the IVD [42, 52, 68, 69, 101, 125, 131, 134-146], their performance has never been validated and they may not always be the correct choice, a result supported by a similar study carried out in rat and rabbit IVD models [151]. To our knowledge, an analysis of reference gene stability was never conducted on porcine IVD and the closest result is an analysis done on porcine articular cartilage, which shows GAPDH as one of the most stable genes [140]. The incongruence between which genes are the best choice for reference is shown to be common; therefore, a stability analysis should be done between tissue types and treatment conditions as it is unknown whether many of these genes are affected by various testing conditions [150, 152]. This analysis indicated that the commonly used reference genes 18s and GAPDH are not necessarily the most optimal selection. Their stability varied between treatment conditions, occupying spots 3 and 4 in the oxygen tension dataset and spots 7 and 5 in the glucose concentration dataset. In fact, YWHAZ and RPL4 were found to be the most stable genes across both datasets and within both datasets.

The T gene is produced mainly in notochordal cells and is indicative of a notochordal phenotype, with other candidate genes being sonic hedgehog and cytokeratin 8 [79, 98, 166-168]. T is highly specific to notochordal cells and is used to track these cells throughout their lifetime [98, 169]. These experiments utilized T to track the stability of the notochordal phenotype, with a decrease in T expression seen with increasing oxygen tension. This may indicate that low oxygen tension preserves the notochordal phenotype. This result is in line with studies in the literature showing that low oxygen tensions help
maintain NP cell phenotype [109, 136, 170, 171]. This was also seen in the morphology of
the cells in the viability images, where most of the cells had large vacuolated morphologies
typical of notochordal NP cells.

Another characteristic of the transition from notochordal NP to chondrocytic NP is
the change in matrix composition, from a highly hydrated proteoglycan matrix with some
elastic collagen II components to a more fibrous and dry matrix based on collagen I [80,
172, 173]. The 1% oxygen tension group was the only group which experienced a
significant increase in collagen II matrix gene expression and was the only group to
maintain no detectable collagen I expression throughout the duration of the experiment.
This indicates that low oxygen levels may be the most optimal for maintaining the
appropriate matrix production of notochordal NP cells. Aggrecan expression was not found
to be altered by the environmental factors presented here. There was a regulatory
transcription factor network found to explicitly resist changes in expression with changes
to oxygen tension which may temper the response of the NP cell to changes in oxygen
tension [136, 174]. Overall, the changes in anabolic gene expression found here show that
low oxygen tension is better for maintaining the phenotypically correct matrix, while high
oxygen tensions lead to a dysregulation of the production collagen species. Other groups
have noted that oxygen tension conditions can have a significant effect on protein
production and matrix regulation of the NP [7, 19, 27, 31, 123, 136, 138, 175, 176].

In contrast, glucose concentration had no significant effect on the expression of NP
cells. Although all of the glucose concentration experiments were carried out at 21%
oxygen tension, only TIMP 3 had an interaction with glucose concentration; there was a downward trend observed for all glucose concentration groups. As there are marked differences between the behavior of chondrocytic and notochordal NP cells [99, 109, 138], and as the native stem cell niche of the NP does not seem to contain cells of notochordal origin, understanding how to maintain the notochordal phenotype is crucial [177]. This is especially the case from a regenerative medicine standpoint as the notochordal NP cells are more effective at producing the correct ECM and they are also able to induce other cells around them to produce the correct ECM [5, 79, 83, 95, 109, 172, 173]. Unfortunately, the notochordal NP cell is not efficient at proliferating, causing long culture times for clinically relevant quantities [90, 91, 177, 178]. The only other gene that showed significant regulation with changes in oxygen tension was MMP 3, which is the most expressed MMP found in the NP [41, 42]. The TIMPs did show an increasing trend with increasing oxygen tension, but no single TIMP showed a significant difference.

Time was the factor that was found to be most influential in the metabolic response of the cells. This is expected behavior for cells adapting to a new environment, but it indicates that a parameter outside this study, such as mechanical loading or a lack of cell-ECM interactions, may be the primary influential factor in determining the behavior of NP cells. Although the timepoints used in this study are longer than the majority of the in vitro experiments within the literature, other groups have found that it can take up to a month to note gross changes in NP matrix composition; thus, even longer timeframes than present in this study would need to be investigated [60, 71, 72, 179]. Additionally, NP cells proliferate slowly and therefore require extended culture times in order to generate cells in
appropriate therapeutic quantities; understanding how to maintain their phenotype will be critical for these therapeutic uses as notochordal NP cells are able to direct the expression of ECM molecules in other cell types under co-culture [6, 83, 91, 177, 178].

Many of the genes investigated here showed significant differences in their expression in response to oxygen tension, glucose concentration, and time, but the list of genes investigated is not exhaustive in terms of disc degeneration. Other groups indicate alterations in expression for other collagen subtypes, such as III, V, VI, IX, and XI, with altered environmental factors[50]. Additionally, other MMPs, such as 7 and 9, have been found around areas of degeneration within the spine [24, 34, 35, 41, 50, 92, 180, 181]. These other genes may still play important roles within the context of disc degeneration, but their overall activity within the NP is lower than the genes investigated within this study [41, 92, 181, 182]. Furthermore, the MMPs are produced in an inactive form and must be first activated before being able to perform their degradation function. This can cause a mismatch between catabolic gene expression and observed catabolic activity. MMP 3, however, is able to act as this pro-protein convertase and initiate MMP activity [37, 41, 46, 92, 180, 181, 183].

The gene expression of the NP cells cultured at different glucose concentrations followed mostly the same pattern as 21% oxygen tension group from the oxygen tension experiments. Again, this seems to indicate that lowered glucose concentrations alone are not enough to significantly alter NP cell behavior. However, glucose has been shown to have a significant effect on NP cells in tangent with another factor [7, 27, 109, 125, 155,
Glucose has been found to be critical in NP cell survival, while other factors, such as oxygen tension and pH, can reach rather extreme values before being detrimental to NP cell survival [7, 26, 27, 109, 184]. NP cells also become less tolerant to these extremes when confronted with low glucose concentrations [7, 26, 109, 125, 155]. The MMPs investigated here preferentially target collagen II followed by collagen I and, taken with the changes in anabolic gene expression, indicate a remodeling of the collagen network of the NP [41, 138]. Low oxygen tension, specifically the 1% group, was found to be the only condition where matrix catabolism and anabolism appeared to be balanced. Other groups have found that low oxygen tension induces cells to produce a more appropriate matrix for the NP niche compared to higher oxygen tensions, which is in agreement with the trends present within this study [123, 138, 175]. The dysregulation of the collagen network is one of the hallmarks of DDD; low oxygen tension provides an apparent protective effect, in addition to showing signs of higher notochordal cell stability [6, 93, 136, 137, 174].

This study found that, in agarose disc culture, oxygen tension was able to significantly influence the gene expression of NP cells, especially over prolonged timeframes. Glucose concentration was less impactful on the behavior of NP cells. Low oxygen tensions were found to confer some protective effect to catabolic trends and dedifferentiation compared to high oxygen tension. The protective role of the TIMP in regulating MMP efficacy was highlighted, though this mechanism might be overwhelmed in degeneration. This study used notochordal cells from a large animal model to gain
insight into the genetic behavior of these cells under altered glucose and oxygen conditions in an agarose gel system over prolonged culture. This provides insight into the beginning stages of degenerative changes, which allows us to find potential treatment targets. It also allows us to optimize culture conditions for future cell therapies and to optimize for such desired output characteristic such as ECM production.
Chapter 5 Modeling Nucleus Pulposus Glucose

Consumption Rate

5.1 Introductory Remarks

We extend the work of the previous two chapters improving from the two-point GCR method to using Michalis-Menten kinetic modeling to determine the value of glucose consumption under varying conditions. Though originally developed to model enzyme reaction kinetics, this method has been used extensively to characterize metabolic rates across a wide variety of cells and conditions, with good agreement between data and model [185]. Though for many situations the two-point method is sufficiently accurate, it suffers from linearization errors, and a large number of experiments are needed to determine the consumption rate at various concentrations. The Michalis-Menten method fits the data to the equation:

\[ v = \frac{V_{\text{max}}[S]}{K_m + [S]} \]  

(5-1)

where \(v\) is the reaction velocity (i.e., glucose consumption rate, GCR), \(V_{\text{max}}\) is the maximum reaction rate, \(K_m\) is the Michalis constant and represents the substrate concentration at which \(v=V_{\text{max}}/2\), and \([S]\) is the substrate (i.e., glucose). Once \(V_{\text{max}}\) and \(K_m\) are characterized, it is possible to describe the rate at any substrate concentration and make predictions with high fidelity. As such, it allows for much higher predictive power which would be useful in future modeling efforts as well as culturing the cells.
In addition to determining GCR, during this set of experiments the lactate assay was performed in order to determine the lactate production rate (LPR). Since the NP cells, both notochordal and chondrocytic, rely on glycolysis for the bulk of their energy production, lactate must be produced in large quantities [7, 27, 110, 112]. The NP LPR is poorly characterized, with studies finding positive, negative, and no Pasteur effects; Pasteur effect is an increase in the glycolysis rate (hence increasing the LPR) under anaerobic conditions, negative Pasteur effect, or no effect [7, 27, 28, 32, 33, 186]. Some of these differences could be due to cell source, since Bibby et al. and Cisewski et al. used mature NP cells sources, while Naqvi et al. used a more notochordal source [7, 27, 32, 109]. Knowing both the glucose consumption and the lactate production rates can give a good insight into the cell’s energy balance. Though lactic acid could build up to levels that would affect the cells, in situ experiments and simulations indicate that this is generally not the case [20, 21, 111, 112, 184]. Very few studies have reported glucose consumption in terms of Michalis-Menten kinetics, even less have used notochordal cell types [7, 27, 28, 31, 109, 110, 125].

Again, during this experiment RT-PCR will be used to evaluate a panel of genes related to maintaining the ECM; the genes can be seen in Table 4. In addition to catabolic and anabolic proteins, three glucose transporters will be monitored as well as the transcription factor HIF-1α.

In this study, we investigated the effect of oxygen tension on the metabolic activity of porcine NP cells. To this end, we measured the GCR, the ratio of LPR to GCR (i.e.,
LPR:GCR), and a panel of genes using RT-PCR in NP cells at varied oxygen tension levels: 2.5% (hypoxia), 5% (normoxia) and 21% (hyperoxia). Normoixa and hypoxia values were chosen based on physiologically relevant levels found in the literature, while hyperoxia is atmospheric conditions [28, 30, 31, 112, 123, 136, 187]. An agarose gel culture system was used to culture fresh porcine NP cells at three oxygen tension levels over a 24-hour adaptation period, which has been shown to be enough time to alter the metabolism of NP cells, as our earlier reported data and other studies show [7, 27, 118, 157, 174]. We then quantified the GCR reaction kinetics based on experimental data. Additionally, we examined the gene expression of the cells over this same adaptation period. Finally, we analyzed the correlation between the HIF-1α transcription factor and the other genes presented within this study. This study provides insight into the metabolism and nutritional demands of nucleus pulposus cells, which is important for understanding IVD tissue pathophysiology, and for harnessing the potential of NP cells in disc regeneration. It also allows for some insights into the genetic regulatory network within the NP which could potentially provide some targets for genetic and biologic based therapies.

5.2 Materials and Methods

5.2.1 Cell Isolation

Cells were harvested and gelled according to previously reported protocols [119-122]. Briefly, Yorkshire pigs (n=5) 4-5 months of age (90-115kgs) were obtained from a local slaughterhouse (Cabrera Farms, Hialeah, FL) and the spine was isolated within two hours
of death. Making transverse cuts through the IVD, the discs were opened and the NP tissue was harvested and placed into an enzymatic solution, composed of high glucose (25 mmol/L) Dulbecco’s Modified Eagle Medium (HG DMEM; Invitrogen Corp., Carlsbad, CA) supplemented with 0.6 mg/mL collagenase (Worthington Biochemical Corp., Lakewood, NJ), 0.6 mg/mL protease (Sigma Chemical, St. Louis, MO), and 25 mM HEPES (Sigma Chemical, St. Louis, MO), to digest the tissue and release cells. Digestion occurred for 24 hours under continuous agitation in an incubator at 37°C, 5% CO2. After enzymatic digestion, the solution was filtered through a 70-µm cell strainer (BD Biosciences, Bedford, MA). The suspension was then diluted, centrifuged, and resuspended to a concentration of 2x10^7 cells/mL in HG DMEM supplemented with 10% fetal bovine serum (FBS) (Atlanta Biologicals, Flowery Branch, GA) and 1% Antibiotic Antimyotic (AA) (Atlanta Biologicals). The suspension was then mixed with 4% agarose (Sigma Chemical), cast into custom molds (discs with d=8 mm) with 100µL per construct, and allowed to solidify, bringing the final constructs to 2% agarose gels containing 1x10^6 cells. The cells were then cultured 48 hours in 5mmol DMEM with 10% FBS, 1% AA, and 25 mM HEPES at 37°C, 5% CO2, 5% O2.

5.2.2 Glucose Consumption Rate Studies

Following the 48-hour incubation, the gels were divided into three groups depending on oxygen tension level: 2.5% (hypoxia), 5% (normoxia) and 21% (hyperoxia). Groups were cultured in 5mmol DMEM containing 10% FBS, 1% AA, and 25mM HEPES at 21% O2, 5% O2, or 2.5% O2 and 37°C, 5% CO2. 5% (n=20) oxygen tension groups were tested immediately (i.e., no additional adaptation period was necessary) while 2.5% (n=21) and
21% (n=20) groups were given a 24-hour acclimation period before proceeding with glucose consumption rate studies. Each hour, 1 μL of media was collected from each sample and glucose concentrations were measured using a custom modified glucose meter made from a commercially available blood glucose monitor and testing strips (Accu-Chek Aviva, Roche Diagnostics, Indianapolis, IN) and a sourcemeter (Model 2400, Keithley Instruments Inc., Cleveland, OH) to achieve high fidelity glucose measurements. Data were collected using a custom LabVIEW (National Instruments Corp., Austin, TX) script. Final cell viability was assessed using Live/Dead™ (Sigma Chemical) staining to ensure viability was kept above 90% (data not shown). A brief graphical overview of the experiment can be found in Figure 5-1.

The GCR was described using Michalis-Menten kinetics:

$$\frac{dC_{gluc}}{dt} = -R_{gluc} = -\frac{V_{max}C_{gluc}}{K_m + C_{gluc} \rho_{cell}}$$

(5-2)

where $C_{gluc}$ is the glucose concentration (mM), $t$ is the reaction time (hr), $V_{max}$ is the maximum consumption rate (nmol/million cells/hr), $K_m$ is the Michalis constant (mM), and $\rho_{cell}$ is the cell density of the well (million cells/mL).

The glucose concentration data was curve-fit to the analytical solution to the Michaelis-Menten equation using the curve fitting function of the SciPy (v 0.18.1) library using Python (v 3.5.2) in order to determine the kinetic coefficients $V_{max}$ and $K_m$.:
where $C_0$ is the initial concentration, and $t_0$ is the starting time.

Figure 5-1: Experimental overview of Michaalis-Menten GCR experiment
5.2.3 Lactate Assay

The lactate assay was performed using previously reported protocols [110]. Briefly, an assay mix was made by combining 5mg/mL β-Nicotinamide Adenine Dinucleotide (Sigma) and 22.25 units/mL L-Lactic Dehydrogenase (Sigma) in a 0.2M glycine buffer (Sigma). A standard curve of (1000µM-62.5µM) was made using experimental media and Sodium L-Lactate (Sigma). The assay mix and samples were combined in equal parts in a 96-well plate. The absorbance at 340nm was then measured using a plate reader (SpectraMax M2e, Molecular Devices, Sunnyvale, California).

The ratio of lactate production rate to glucose consumption rate (LPR:GCR) was calculated by:

\[
LPR: GCR = \frac{C_l(t)}{C_g(t_o) - C_g(t)} \tag{5-4}
\]

where \(C_l(t)\) is the final concentration of lactate in the media, and \(C_g(t_o)\) and \(C_g(t)\) are the initial and final concentrations of glucose, respectively, measured as described above.

5.2.4 RNA Extraction and RT-PCR

Total RNA was extracted using TRIzol (Tri-Reagent, Molecular Research Center, Cincinnati, OH) protocol. Total RNA was quantified using the Qubit RNA BR assay kit (Life Technologies, Carlsbad, CA) and reverse transcribed to cDNA using the High Capacity cDNA reverse transcription kit (Applied Biosystems, Foster, CA), according to manufacturers’ specifications. Gene expression was quantified through the use of real time-PCR (One step Plus, Applied Biosystems); a list of the genes studied can be found in Table
4. The values were transformed to ΔCq values using the NormqPCR package (1.7.1) in R (3.3.2).

Table 5-1: Panel of genes assessed. List includes anabolic, catabolic, inhibitors of catabolism, glucose transporters, the HIF 1α transcription factor, and the notochordal marker Brachyury. GAPDH is used as the reference gene.

<table>
<thead>
<tr>
<th>Gene</th>
<th>Forward Sequence</th>
<th>Reverse Sequence</th>
<th>Target Gene</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAPDH</td>
<td>GTTTGTGATGGGCCGGTGAACC</td>
<td>AGCTTGACGAAGTGGTGT</td>
<td>NM_001206359.1</td>
</tr>
<tr>
<td>TIMP 1</td>
<td>ATTTGTTGGGAGCCCCAGAGT</td>
<td>CCAAGGGCTATGACCCCT</td>
<td>NM_213857.1</td>
</tr>
<tr>
<td>TIMP 2</td>
<td>CTACCGCAACCCCATCAAC</td>
<td>ACGAGGGGGCCGCTGAGAT</td>
<td>NM_001145985.1</td>
</tr>
<tr>
<td>TIMP 3</td>
<td>CCTGCCCTGCTTGGTGTGACCT</td>
<td>CGGATGACGGCGTGTGTTTT</td>
<td>NM_003126073.4</td>
</tr>
<tr>
<td>MT 1</td>
<td>CTGGACTGGTCCGAAATGAGG</td>
<td>AGGGGTCATGGAGGTCTCA</td>
<td>NM_214239.1</td>
</tr>
<tr>
<td>MMP 1</td>
<td>GGACCTGAGGAAACCTTGGCT</td>
<td>GCCTGGATGCACTCAGATC</td>
<td>NM_001166229.1</td>
</tr>
<tr>
<td>MMP 3</td>
<td>TCCCTAGTTGTTACTTACGCCAC</td>
<td>TTAGCAATCTCTGAAGTGAAGTGCTT</td>
<td>NM_001166308.1</td>
</tr>
<tr>
<td>MMP 13</td>
<td>CATGAGTTGCGACATATTT</td>
<td>GTGGCCTTTGCCAGTGTAG</td>
<td>NM_001166523.1</td>
</tr>
<tr>
<td>Aggrecan</td>
<td>CAGGCTGGACTGGTCTGACGAC</td>
<td>GTGAGTCATGAGGTCTTGT</td>
<td>NM_001166523.1</td>
</tr>
<tr>
<td>Collagen I</td>
<td>TGACGTGATCTGCGACGAAA</td>
<td>GCACCATCTTTCCAGACGAC</td>
<td>NM_005668927.1</td>
</tr>
<tr>
<td>Collagen II</td>
<td>CACGGATGGTCCCAAAGG</td>
<td>TCGGGCTTTTCACCAACT</td>
<td>XM_005668927.1</td>
</tr>
<tr>
<td>HIF-1α</td>
<td>TGTACCTTAACTAGCAGGGGA</td>
<td>AAATTGAGCGGCCCAGAAGT</td>
<td>NM_001123124.1</td>
</tr>
<tr>
<td>GLUT-1</td>
<td>GCCCCATCCTCCAGAGGGA</td>
<td>AAATTGAGCGGCCCAGAAGT</td>
<td>NM_001123124.1</td>
</tr>
<tr>
<td>GLUT-3</td>
<td>AAAGTCCTGGGTCCCGCTCGT</td>
<td>ATTCCTGGGTGCCCACAATC</td>
<td>XM_013988404.1</td>
</tr>
<tr>
<td>GLUT-9</td>
<td>TCTGGCTTTGTCATGGACG</td>
<td>GAAGGGTACATGCCAAAA</td>
<td>XM_013991444.1</td>
</tr>
</tbody>
</table>

5.2.5 Statistical Analysis

The differences between oxygen tension groups, 2.5% (n_gels=21), 5% (n_gels=20), and 21% (n_gels=20), for Km, Vmax, LPR:GCR, and gene expression were analyzed using the base R (v 3.6.1) one–way ANOVA, with a blocking factor around the pig that generated the gels (n_pig=5), the R model can be found in Equation 5-1. The significance level was set at p<0.05. Values for Km, Vmax, and LPR:GCR are given as mean ± standard error of the mean (SEM). Gene expression is shown as boxplots of -ΔCq values.
5.3 Results

In this study, the glucose consumption rate was measured at three oxygen tension levels (2.5%, 5%, 21%) in a total of 61 samples from five pigs. The values of $V_{\text{max}}$ and $K_{\text{m}}$ were determined through theoretical curve-fitting of the analytical solution of the Michaelis-Menten equation. A sample plot of a theoretical curve-fit is shown in Figure 5-2. There was very good agreement between the experimental data and theoretical curve fitting ($r^2=0.977\pm0.026$, n=61). The results for $V_{\text{max}}$ and $K_{\text{m}}$, are shown in Figure 19 and Table 3. There was no significant effect of oxygen tension level on the glucose consumption rate constants for porcine NP cells. The average values for $V_{\text{max}}$ and $K_{\text{m}}$ across all oxygen tension levels were $131 \pm 12 \text{ nmol/million cells/hr}$ and $0.637 \pm 0.102 \text{ mM}$, respectively (n=61).
Figure 5-2: A) Sample curve-fit of GCR experiment. B) Kinteic plots of GCR rates at various glucose concentrations using calculated $K_m$ and $V_{max}$ values.

The ratio of lactate production rate to glucose consumption rate (LPR:GCR) for porcine NP cells was also determined for each sample; the results are shown in Figure 5-3 and
Table 5-2. There was no significant effect of oxygen tension on the LPR:GCR for porcine NP cells. The average LPR:GCR for all groups was 1.91 ±0.08 (n=61).

![Graphs showing Km, Vmax, and LPR:GCR for various oxygen tensions](image)

Figure 5-3: a) Km, b) Vmax, and c) LPR:GCR of the NP cells for the various oxygen tension groups. No significant differences were found between oxygen tension groups.

Table 5-2: Results for Km, Vmax, and LPR:GCR for oxygen tension groups 2.5% (n=21), 5% (n=20), and 21% (n=20), and overall average for all oxygen levels (n=61). All values are given as mean ±SEM.

<table>
<thead>
<tr>
<th>Oxygen Tension</th>
<th>Km (mM)</th>
<th>Vmax (nmol/million cells/hr)</th>
<th>LPR:GCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5% (n=21)</td>
<td>0.608±0.214</td>
<td>127±27</td>
<td>1.98±0.13</td>
</tr>
<tr>
<td>5% (n=20)</td>
<td>0.698±0.095</td>
<td>133±11</td>
<td>1.77±0.16</td>
</tr>
<tr>
<td>21% (n=20)</td>
<td>0.605±0.202</td>
<td>133±22</td>
<td>1.96±0.10</td>
</tr>
<tr>
<td>Average (n=61)</td>
<td>0.637±0.102</td>
<td>131±12</td>
<td>1.91±0.08</td>
</tr>
</tbody>
</table>

No significant differences were observed between oxygen tension groups for any gene as can be seen in Figures 5-4 through 5-8.
Figure 5-4: A) Brachyury and B) Hypoxia Inducible Factor 1α expression in relation to oxygen tension group.

Figure 5-5: Expression of A) aggrecan, B) collagen I, and C) collagen II in response to culture in different oxygen tensions.
Figure 5-6: Expression of A) MMP 1, B) MMP 3, C) MMP 13, and D) MT 1 in response to culture in different oxygen tensions.

Figure 5-7: Expression of A) TIMP 1, B) TIMP 2, and C) TIMP 3 in response to culture in different oxygen tensions.
Figure 5-8: Expression of A) GLUT 1, B) GLUT 3, and C) GLUT 9 in response to culture in different oxygen tensions.

A correlational analysis was used to compare the HIF-1α transcription factor gene expression and the various anabolic, catabolic, TIMPs, and glucose transporters gene expression; the results can be seen in Figure 5-9. All correlations were significant to \( p < 10^{-6} \) and the correlation coefficients were close to or above the 0.84 threshold found to indicate a high probability of co-regulation [188-191].
Figure 5-9: Correlational plots of the panel of genes in the study against the transcription factor HIF 1α. All genes had p values less than 10^{-6} and the correlation coefficients are listed on the plot of the gene. A) The notochordal marker Brachyury. B) The anabolic genes 1) Collagen I, 2) Collagen II, and 3) Aggrecan. C) The catabolic genes 1) MT 1, 2) MMP 1, 3) MMP 3, and 4) MMP 13. D) The inhibitors of catabolism 1) TIMP 1, 2) TIMP 2, and 3) TIMP 3. E) Glucose transporters 1) GLUT 1, 2) GLUT 3, and 3) GLUT 9.
5.4 Discussion

Due to the avascular nature of the IVD, disc cells are exposed to low nutrient levels and steep concentration gradients \textit{in vivo} [6, 10, 11, 192]. This is further exacerbated during degeneration, when reduced endplate permeability and/or diminished vascularity in the adjacent vertebrae leads to a further decrease in nutrient levels [5, 10, 11, 21, 192]. Therefore, disc cells, particularly in the central NP region of the tissue, must survive in low glucose and hypoxic conditions. Cell consumption rates are key determinants of nutrient concentrations in avascular tissues. Cellular activity and viability are dependent on the chemical environment around cells. In particular, previous studies have shown that glucose appears to be the main nutrient necessary for IVD cellular survival [26, 27, 193]. Thus, the objective of this study was to determine the glucose consumption rates by porcine NP cells at several oxygen tension levels.

Only few previous studies, including our study in \textbf{Chapter 3}, have investigated the GCR by cells from the intervertebral disc. Our previous results ranged from 170 nmoles/million cells/hr to 430 nmoles/million cells/hr at oxygen tensions ranging from 1% to 21% [157]. A previous study by Guehring et al. found that the glucose consumption rate by porcine notochordal NP cells was 205 nmoles/million cells/hr at 21% oxygen tension [30], while Naqvi and Buckley [109] found the GCR of porcine NP cells in 5 mM glucose was approximately 110 nmol/million cells/hr at 5% O$_2$, and 40 nmol/million cells/hr at 21% O$_2$. Additionally, the GCR by bovine NP cells ranged from approximately 20 to 120
nmols/million cells/hr, depending on the glucose concentration [7], while that of human nondegenerate NP cells averaged 140 nmol/million cells/hr [32]. Our average value for glucose consumption rate at 5 mM glucose (116 nmol/million cells/hr, calculated based on average $V_{\text{max}}$ and $K_m$ values from Table 1), is in agreement with these earlier studies. The range of GCR values in the literature may be due to differences in cell population (e.g., notochordal vs. mature/chondrocyte-like), cell isolation techniques, cell culture conditions, and/or variation between individual animals. The $K_m$ of the glucose transporters, GLUTs 1, 3, and 9, was found to be much higher than the glucose consumption rate and indicates that transport of glucose across the membrane is not the limiting factor in glucose utilization by the NP cell [194-198]. We found that glucose concentration did not significantly affect the GCR of NP cells in our previous study, though there is a clear trend seen in this study with lower glucose concentrations having lower GCRs. The lowest glucose concentration was 1mM in our earlier study, which is significantly higher than the $K_m$ of 0.6 mM. Thus, our earlier work fell with the glucose concentration range within the asymptotic region of the GCR curve where even large changes in glucose concentration produce small changes in GCR.

Our results did not show any significant effect of oxygen tension level on the glucose consumption rate by NP cells, like our previous studies which had timepoints of a similar time frame, even though other groups were able to see significant metabolic changes in similar time frames [7, 27, 118, 125, 157, 174]. Several studies have
investigated the oxygen tension effect on IVD and chondrocyte glucose consumption rates, with mixed results. Various studies have found a positive Pasteur effect (i.e., increase in the glucose consumption rate with a decrease in oxygen tension) [28, 31], a negative Pasteur effect [7], or no Pasteur effect at all [27, 32, 33]. Differing trends may be due to variations in culture conditions, including differences in isolated cells, cells expanded in monolayer, vs tissue explants. It has been suggested that glycolytic rates at low glucose concentrations may be influenced by the extracellular chemical environment along with cell-matrix interactions [7], which are not present for isolated cells as in this study. Though not explored in this investigation, the effects of pH, glucose, lactate, osmolarity, and mechanical loading have been previously investigated and found to have an influence on the metabolism of the NP [30, 32, 33, 110, 120-122, 125, 143, 164, 192, 193]. Though many of those factors have an effect, most are modest (especially over shorter adaptation periods) [120, 125] and many of the experiments contained groups with conditions which are not found or predicted physiologically, even in degenerate discs [21, 124, 186, 199]. Moreover, with the use of the HEPES buffer system and using the same media formulation for all experimental conditions should keep the pH and osmolarity stable for the duration of the experiment. Most recently, Cisewski et al [32] found that oxygen tension significantly affects the glucose consumption rate of isolated nondegenerate human NP cells: the consumption rate at 5 mM glucose levels was highest at 5% oxygen tension, and decreased for both 1% and 21% O₂. This is similar to the trend seen here, see Figure 5-3: at 5 mM glucose concentration, the consumption rate is highest for the normoxia group, and decreases in both hypoxia and hyperoxia. However, as noted, there was no statistically
significant effect of oxygen tension on GCR in this study, which may be due to low sample size and high variability between animals. Additionally, this may be due the short adaptation period during which cells adapt to the oxygen environment. In this study, cells were acclimated to hyperoxia (21%) or hypoxia (2.5%) for 24 hours prior to the experiment, which may not be long enough to alter gross cellular behavior to this particular metabolite at these timescales [157]. Nonetheless, the quantitative values are in agreement with the literature, and the overall trend may also be explained by the specific behavior and/or adaptation of NP cells.

Numerous studies have investigated the effects of oxygen tension level on the activity and viability of IVD NP cells. Neidlinger-Wilke et al [200] previously showed that varying the oxygen tension for bovine NP cells in alginate culture had on a minor influence on expression of important anabolic and catabolic genes, while Risbud et al [176] found that hypoxia led to an upregulation of key phenotypic markers (e.g., SOX-9, collagen Type II, aggrecan) in rat NP cells in monolayer culture. Mwale et al [123] also found that varying oxygen levels from 1% to 21% had little effect on GAG production by bovine NP cells in alginate culture. Previous studies have found that exposing IVD cells to hypoxia alone did not alter their growth or survival in monolayer or alginate culture [156]. Taken together, these results suggest that NP cells may be specifically adapted to not respond to a change in oxygen environment (particularly hypoxia) as readily as other cell types, which is in agreement with the results of this study.
We also determined the ratio of lactate production rate to glucose consumption rate for the porcine NP cells in this study. IVD cells are known to rely mainly on glycolysis (i.e., anaerobic respiration) in order to produce necessary energy for cellular processes [28, 31]. For pure glycolysis, the LPR:GCR should be 2.0, assuming 2 moles of lactate are produced for each mole of glucose consumed. The average LPR:GCR found in this study (1.91±0.08) was close to this theoretical value, see Table 1 and Figure 2. Although no significant difference was found for different oxygen tension levels investigated, the trend showed that this ratio was slightly lower for the 5% O₂ group (1.77±0.16), which suggests that glucose is used for additional reactions other than those resulting in the reduction of pyruvate to lactate. Our results are in agreement with previous studies in the literature, which ranged from 1.7 for notochordal NP cells, to 2.1 for mature NP cells [30], and 2.01 for bovine NP cells [7]. Knowledge of the specific ratio is important as it provides key insight into the metabolic pathways of disc cells. Furthermore, an LPR:GCR of 2 is often assumed, particularly in computational modeling of the disc. Thus, precise information on this ratio is important for accurately predicting the in vivo nutritional environment in the disc.

In this study, we modeled glucose consumption rate reaction kinetics by the Michaelis-Menten equation. In general, most previous studies have determined the averaged GCR by calculating the total glucose consumed over a given time period (i.e., two-point method). Only few studies have used the Michaelis-Menten relationship to describe glucose consumption by IVD cells and chondrocytes. In investigating the lactate production rate, Guehring et al. found that the value of $K_m$ for notochordal NP cells was
0.199 mM, compared with our averaged value of 0.637 mM [30]. Additionally, a study of articular chondrocytes found the value of $K_m$ to be 0.35 mM [201]. Differences in values determined for $K_m$ may be due to the techniques used. In the method described here, concentration values were measured every sixty minutes; however, more frequent measurements may refine the value for $K_m$, though this time frame was found to be the optimal solution for providing enough timepoints for a good fit, without disturbing incubator and media oxygen concentrations. The general trend of a decrease in GCR at lower glucose levels has also been previously reported by Bibby et al [7], who found that glucose consumption by bovine NP decreased rapidly below 1mM glucose concentrations, which is in agreement with our findings. This general behavior may be due to the mechanism of glucose transport into the cell via the GLUT transporter: at higher concentrations of glucose, these GLUT transporters may be saturated, thereby limiting the GCR. Similar trends have been seen in articular chondrocytes [201].

None of the genes examined were found to have differential expression within the time frame of this experiment. This result is not unexpected as some of the genes did not show significant changes in previous experiments, even after 10 days in culture [157]. Moreover, most genes showed significant changes only after several days in culture, indicating that, for this panel of genes, the adaptation window may be too small to generate an appreciable effect. Though there were no noted differences in gene expression in the shorter adaptation time of this experiment, previous studies have shown that 24 hours is enough time to observe changes in NP cell behavior [7, 27, 118, 125, 174]. Moreover,
oxygen tension has been found to have a significant influence on protein production, gene expression, and other metabolic activities of the NP [7, 31, 109, 123, 124, 157, 174].

Gene expression data should be tempered with the fact that many genes exhibit post transcriptional and post translational modifications and control, ranging from alternate splicing to silencing by microRNA interactions to requiring activation by another protein [42, 202, 203]. Therefore, although the gene expression showed no significant changes in expression, there still may be some effect on the final behavior of the NP cell and further protein quantification studies would be required. Additionally, this panel of genes, though extensive for RT-PCR profiling, lacks the breadth that microarray experiments are able to perform, though they provided a more detailed picture of differences in the specific genes interrogated. Thus, although these particular genes showed no significant differences, there may be a host of other genes which control the final production of many ECM molecules which are tangentially related but were not assessed in this experiment.

The transcription factor HIF-1α is known to be stable across multiple oxygen tension levels in the NP, even though normally it is activated only under hypoxic conditions in other cell phenotypes [29, 92, 136, 204]. Moreover, HIF-1α is only natively expressed in the NP of the IVD, while AF and CEP cells show no such expression; therefore, it can potentially be used to further identify the cells [92]. All correlations with HIF-1α were significant to $p<10^{-6}$ with correlation coefficients close to or above the 0.84 threshold. This indicates a probability of around 50% that the transcription factor is involved in the regulation of the target gene or that this transcription factor and the gene share another
transcription factor in common [189-191, 205]. One potential candidate is the Cited2 gene that was found to modulate the HIF-1α/VEGF response in rat IVD, though HIF-1α is itself a transcription factor normally involved in regulating cell response to hypoxic conditions [29, 136, 170, 171, 174, 187]. These results provide an excellent foundation for knockdown and overexpression studies to examine the complexities of how HIF-1α regulates or is regulated in terms of matrix remodeling within the IVD. Interestingly, the notochordal marker Brachyury was also correlated to HIF-1α at the ρ=0.84 level. HIF-1α has been found to regulate numerous other genes and transcription factors in the IVD and appears to be critical in maintaining the NP cell phenotype [92, 136, 170, 171]. These results hint at the regulatory network present within the NP cell population which controls ECM production and metabolism.

This study only altered oxygen tension conditions of NP cells, which by itself may not be enough to cause significant changes to the behavior of the IVD cells. Other investigators examined the behavior of the IVD cells under coupled conditions, such as low glucose and mechanical stimulation or low oxygen and low pH, and found the response to be much greater than when only confronted with one adverse condition [7, 42, 72, 109, 123]. Bibby et al., in particular, assessed a range of differing nutrient conditions and was able to model NP cell metabolic responses roughly to the coupling of glucose concentration, pH, and oxygen tension in which all three nutrient levels were influential in determining the response of the NP cells [7]. Also, though kept constant between groups and kept within physiological limits, the effect of osmolarity on NP cell behavior was not studied, which is another factor that has been found to be important in the regulation of the
NP cell metabolism and ECM production [65, 146, 186]. Though it ultimately complicates final experiment design, the NP cells of the IVD seem to be able to withstand adverse conditions from a singular source, but start to experience adverse reactions once those hostile conditions are coupled.

Computational modeling is often used to investigate the nutritional environment in the avascular IVD, given that in vivo measurements of nutrient concentrations have proven difficult. Nutrient concentrations depend on both solute transport rates (i.e., diffusivities) as well as rates of cellular consumption. Therefore, in order to accurately predict the nutrient environment via a mathematical model, quantitative relationships relating the nutrient consumption rates with nutrient levels are crucial. Here, we have established a mathematical relationship between the glucose consumption rate and the local glucose concentration for NP cells. This relationship is of particular importance given that glucose is believed to be the main nutrient essential for disc cell survival [26, 27, 193]. Therefore, the ability to precisely predict glucose levels in the tissue is crucial for understanding the effects of the nutritional environment on cell activity and survival, and related tissue degeneration. Of note, we have only investigated the effects of oxygen tension and glucose concentration on the GCR of NP cells; however metabolic rates may also depend on other factors in the chemical microenvironment, including pH and osmolarity. For instance, Bibby et al [7] previously showed that bovine NP metabolic rates depend on pH but not osmolarity in alginate culture. Therefore, further investigation should be taken to determine how these other physiologically relevant factors may contribute to cellular metabolism.
In this study, we used cells harvested from porcine spines, which represent a notochordal cell population. The NP of human IVD are known to lose their notochordal cells early in maturation [6], though there is still some debate as to whether the chondrocytic NP cells of the adult human IVD are notochordal in origin or migrate in from the CEP [98]. In addition the transition from notochordal to chondrocytic NP cells in the IVD corresponds to the first signs of degeneration [6]. Both cell populations produce and integrate ECM in a manner that is unique from the other [95]. It has also been shown that notochordal cells are more metabolically active than chondrocytic NP cells [30]; as such, the results of this study may vary from values for mature and/or degenerated human IVD cells. These properties alone would make notochordal cells an attractive target for cell therapies, though the higher metabolism of the notochordal cells would need to be accounted for, but in addition to direct matrix synthesis notochordal cells have been shown to influence the matrix synthesis of chondrocytic cells, even in fairly low cell ratios [83, 93, 118]. In addition notochordal cells also appear to influence the migration of chondrocytic NP cells [94]. All of these properties taken together make notochordal cells an attractive source for future regenerative therapies, and they could be beneficial even in small numbers. Therefore, there is a need to understand cellular metabolism and nutritional demands of these cells in order to harness their potential in IVD regeneration techniques. HIF-1α, an important hypoxic transcription factor was found to be strongly correlated with the expression of many of the genes interrogated within this study hinting at it importance of the regulatory network of the NP cell metabolism [29, 136, 170, 171, 187, 189, 205].
This could potential lead to target of gene therapy and biologics treatments that target this regulatory network.

In summary, we have quantified the GCR reaction kinetics and LPR:GCR ratio of porcine NP cells at several oxygen tension levels. Because we monitored the glucose concentration at several time points, we were able to describe the glucose concentration dependence of GCR using Michaelis-Menten reaction kinetics, with values for $V_{\text{max}}$ and $K_m$ determined via curve-fitting experimental data. Our results did not show any significant effect of oxygen tension level on NP cell GCR, which is in agreement with some studies in the literature [27, 32, 33]. No significant differences were found between oxygen tension groups for any of the genes investigated. The adaption window may have been too short to register appreciable differences. Moreover, many genes were found to be highly correlated to the expression of the HIF-1α transcription factor, strongly suggesting its involvement in the regulation of NP cell behavior. The results of this study are important for understanding the nutritional demand of NP cells, which can provide key insight into tissue pathophysiology as well as optimal conditions for NP cell survival for tissue regeneration approaches. Additionally, the quantitative relationships determined here are crucial for accurate computational modeling of the IVD, which can help to predict the *in vivo* tissue environment. Though no changes in gene expression were found, potential targets for further genetic studies and therapeutics candidates were identified.
6.1 Introductory Remarks

As mentioned in previous sections, the response of IVD cells is complex and responds to many environmental cues [7, 31, 124, 146, 156]. Frequently, it takes more than one of the critical nutrients to be out of balance in order to bring about adverse changes to the cells of the IVD [7, 31, 124, 146, 156]. Though numerous studies have investigated many of these properties \textit{in vitro} using monolayer culture or embedding the cells in a hydrogel [33, 64, 109, 138, 206], there are many cues that the cells receive from the native ECM, which can also influence the behavior of the cells [62, 207]. A potential solution is to use \textit{in vivo} studies [69, 77, 108, 112, 134, 208], with the drawback of having less control over input conditions of the disc. Many investigators have developed bioreactors and the associated systems in order to bridge the gap between providing the control of an \textit{in vitro} study while still providing a mostly \textit{in vivo} environment [58-61, 70-72, 84, 124, 155, 209-213]. This allows the cells of the disc to be left \textit{in situ} surrounded by the native ECM while controlling nutrition and loading on the disc [70-72, 84, 124, 155, 209-212].

Large animal models are favored in the literature due to their similarity in size and shape to human IVD, thereby providing similar loading characteristics and diffusional concerns that small animal models cannot replicate [74, 77, 78, 212-214]. Bovine, ovine, and porcine IVD sources are generally selected due to ease of access to large number of discs through either an abattoir or their more common use in other institutional research
These three sources are closely matched in size to each other and possess discs of a similar scale and shape to that of human IVD.

Because of the avascular nature of the disc, there is no need to perfuse the tissue, but nutritional considerations are still paramount [70-72, 84, 124, 155, 209-213]. Because the endplate route is the primary pathway through which nutrients diffuse into the NP [17, 107, 108], providing unobstructed diffusion across this boundary becomes one of the primary concerns in most whole organ culture systems of the IVD [71, 72, 84, 155, 209-213]. Thus, preparation of the IVD becomes a critical. The two most common preparations are whole IVDs capped with 1-2mm of vertebral bone (bone in) or to shave off the bone and leave only non-ossified endplate (boneless). A third possible option is to also remove the endplates as well, but this is far less common due to the accelerated deterioration of specimen integrity [66, 67, 213]. Though the boneless procedure has less diffusion concerns, most bioreactors which apply loading to IVD segments use the bone-in preparation for structural stability [66, 67, 213].

Further expanding on nutritional concerns, media handling is an important aspect of bioreactor design [56, 57, 66, 67, 70, 71]. The simplest of these systems is the batch reactor design where there is no explicit media handling and the IVD would simply be submerged in a media bath that would either be complete enough to run the course of the experiment or would need to be periodically changed. This design is not favored in the literature due to the aforementioned diffusion concerns, especially considering that the endplate is both the main site of diffusion for the NP and the location of force transduction
into the disc [17, 19, 66, 67, 70, 71, 107, 209, 214]. The use of porous plates thus is common, as they allow media to move across the interface while still being sturdy enough to apply the desired forces without breaking down [66, 67, 70, 71]. Furthermore, using a batch system with no media handling or circulation would create concentration gradients due to the stagnant nature of the media and may be a source of error in applying specified nutrient conditions to the boundary of the IVD. Thus, most of the bioreactors in the literature have some form of media handling capability that pumps the fluid in a controlled way to ensure good mixing and limiting and diffusion gradients caused by static fluid [66, 67, 70, 71, 209].

Since it has been shown that cells of the IVD respond to mechanical loading, this factor is critical when designing a whole organ system [67, 69-72, 127, 142, 145, 212]. Even if mechanical load is not the parameter of interest, mechanical confinement must be at least considered as free swelling IVDs can gain 24% of their initial height while unloaded in saline or PBS solutions [66, 67, 210]. Additionally, some form of cyclical or diurnal loading, within normal physiological limits, has been found to promote cell viability [127]. Though free swelling discs can maintain high cell viability up to 14 days, the lack of mechanical loading and the change in osmotic environment are undesirable [66, 67]. Most bioreactors in the literature provide axial loading on the disc, due to ease of design and implementation; additionally, one study used a bioreactor to induce torsion, while none included bending of the IVD segment [66, 67, 70-72, 209, 212].
Thus, the goal of this study was to develop a whole IVD bioreactor and use it to probe the effect of simulated physiological loading as well as oxygen tension in the gene expression and tissue properties of whole IVD segments. The discs were assessed at 7 and 14 days in whole organ culture and compared to their harvest values. Gene expression of key transcription factors as well as catabolic genes and their inhibitors were monitored in order to gain insight into the remodeling process of the IVD under varied oxygen and loading conditions over extended periods. These results can be used to find gene therapy targets and to help optimize culture conditions to expand IVD cells for future cell-based therapies. The whole organ culture system allows in vitro control while presenting native ECM.

6.2 Materials and Methods

6.2.1 Design of Intervertebral Disc Bioreactor

A custom intervertebral disc bioreactor was designed and validated, which can be seen in Figure 6-2. The device consists of aluminum support structures where no liquid contact is expected, while stainless steel is used anywhere where media or tissue will interface with the device. All major load bearing components were first modeled using SolidWorks CAD software, technical drawings and bill of materials (BoM) can be found in Appendix A. A finite element model analysis was run on the assembly using the FEM features of SolidWorks to ensure all mechanical components would be able to sustain the loads placed on it. Aluminum and stainless-steel stock was purchased from McMaster-Carr (McMaster-Carr, Elmhurst IL). Loading was achieved using a linear actuator (Windy Nation, Ventura,
controlled by an Arduino MEGA 2560 (Arduino, Turin, IT) microcontroller. Segment loading was monitored by a load cell (Phidgets, Calgary, AB), mounted in line with the linear actuator. Custom motor controller and custom analog to digital (ADC) boards were designed using KiCAD (Cern, Geneva, SW) design software, schematics, board layouts, and BoM can be found in Appendix A. Custom software, found in Appendix B, was written for running the microcontroller as well as interfacing with the device for calibration and active monitoring. Loading was administered to the IVD segments through the use of custom 3D printed stainless steel porous platens (J&J 3D Printing Center of Excellence, Coral Gables, FL). A custom analog pump system maintained fresh media exchange with an external reservoir.

The bioreactor had several features enabled by the microcontroller and supporting hardware. The main function was to read the load on the disc and adjust the linear actuator output to the proper level. This was achieved by using proportional/integral (the PI from PID) control loop scheme. The next task of the bioreactor was to record the setpoint and actual loading and record these data to an SD card for later evaluation; to do this, an existing module-based solution was used. The final task of independent timekeeping was handled through the use of a real-time clock (RTC) module. A schematic representation of this process can be found in Figure 6-1.
Figure 6-1: Pictographic diagram of the IVD bioreactor. An analog signal from the load cell, made in the
whetstone bridge configuration, was read into an ADC on the sensor board and the signal was digitized
\((f_{ADC})\). This signal was then sent back to the Arduino, which read a real-time clock to adjust the setpoint
\((f_{set})\). Setpoint data and load data were then fed into a custom PI algorithm used to set output PWM signal
\((f_{PID})\). Load data, setpoint data, and time were all recorded to an SD card during operation. The PWM
signal from the Arduino was sent to the motor controller where it was converted to an analog current signal to
drive the motors \((f_{DAC})\).
Figure 6-2: Whole IVD bioreactor. Reactor on left does not have the incubation vessel to properly show internal components.
6.2.2 Preparation of Intervertebral Disc Segments

Intervertebral discs from five Yorkshire pigs, which were approximately 4-5 months of age and 45-55kgs in weight, sourced from a local slaughterhouse (Cabrera Farms, Hialeah, FL) were harvested. Discs from the L1 to the S1 vertebrae were isolated and cut adjacently to the endplate, preserving the bone. Pictures were then taken of the disc against a calibrated background, as can be seen in Figure 6-3. Disc cross sectional area was then calculated using ImageJ (NIH ImageJ, Rockville, MD) software. One disc was used as the control; the NP and AF tissues were harvested immediately (i.e., day 0 timepoint). The other discs were inserted into bioreactor vessels and loaded statically at 0.2MPa for 16 hours a day and a cyclic 0.4MPa-0.8MPa for 8 hours a day at 0.5 Hz to mimic a diurnal loading pattern, a schematic representation of these parameters can be found in Figure 6-4. Discs from the thoracic region, TH5 to L1, were also harvested with bone-in cuts and allowed to incubate under free-swelling conditions. Free-swelling discs were harvested for viability, ECM, and gene expression analysis at days 1, 7, and 14. Discs placed in the bioreactor were harvested at days 7 and 14 for viability, ECM, and gene expression analysis, an overview of the experiment can be found in Figure 6-3. For all discs, media was exchanged at day 7.
Figure 6-3: Experimental overview of bioreactor experiment

- **Parameters**
  - 0.6 MPa ± 0.2 MPa
  - @ 0.5 Hz
  - 0.2 MPa

- **8 hr**
- **16 hr**

- **Day 0**
  - Disc Harvest
  - Free-Swelling
  - Bioreactor
  - 21% O₂
  - 5% O₂
  - 21% O₂
  - 5% O₂

- **Day 1**
  - Measure Disc
  - Measure Disc
  - Measure Disc

- **Day 7**
  - Measure Disc
  - Measure Disc

- **Day 14**
  - Measure Disc
  - Measure Disc

- **Measurements**
  - GAG
  - Hydroxyproline
  - Gene Expression

- **Media Change**
Figure 6-4: Image of cut IVD segment on calibrated background. Each square has a 1 cm length.
Figure 6-5: Loading pattern placed on IVD within the bioreactor. The discs were subject to 8 hours of 0.6MPa ±0.2MPa dynamic loading followed by 16 hours of static compression at 0.2MPa to mimic a diurnal loading pattern.

6.2.3 Viability Testing of Intervertebral Disc Segments

Viability was assessed using Hoechst 33342 dye (Thermofischer, Waltham, MA) and Propidium Iodide (Sigma) according to previous protocols [215]. Briefly, a solution of 10µM Hoechst and 5µM Propidium Iodide in PBS was made. 100-300mg tissue samples of either NP or AF were placed in 1mL of staining solution and placed on a shaking incubator at 4°C for 30min. The staining solution was then removed and replaced with fresh PBS and the tissues were then incubated for another 30min at 37°C. Tissues were then imaged under a fluorescent microscope (microscope: Motic AE31E, Richmond, BC; camera: Zeiss AxioCam IC1, Oberkochen, GER). Hoechst dye was used to stain all nuclear material while Propidium Iodide was used to stain dead cells. Live cells were all cells were
Hoechst dye was present but Propidium Iodide was absent, while dead cells were cells where Propidium Iodide was present.

6.2.4 Extracellular Matrix Characterization

GAG content was assessed using the 1,9-dimethylmethylene blue (DMMB) (Polysciences Inc., Warrington, PA) assay [216]. Briefly, tissue specimens were harvested and weighed. Specimens were then lyophilized overnight and then their dry weights were recorded. Lyophilized samples were then digested in 250µg/mL papain (Sigma) solution, at a concentration of 1mL of every 40mg tissue dry weight, at 60°C overnight. A standard curve was prepared using chondroitin A standard (Sigma). Tissue digest was then mixed with DMMB solution and read in a microplate reader (Molecular Devices SpectraMax M2 Series, Sunnyvale, CA) at 525nm.

Collagen content was analyzed through the use of the hydroxyproline assay [217, 218]. Briefly, the tissue digests from the GAG content measurement were mixed with an equal amount hydrochloric acid and incubated overnight at 105°C. Next, isopropanol was added and the solution was vortexed. Then, a Chloramine-T solution was added and solution was vortexed and then incubated for 5 minutes at room temperature. Following this, Ehrlich’s reagent was added and the solution was once again vortexed and then allowed to incubate at 60°C for 20 minutes. After incubation, the solution was rapidly cooled to room temperature by using running water. A standard curve was prepared using hydroxy-l-proline standard (Sigma) The absorbance was read at 558nm in a microplate reader in a microplate reader.
6.2.5 Gene Expression

Total RNA was extracted using TRIZol (Tri-Reagent, Molecular Research Center, Cincinnati, OH) protocol. Total RNA was quantified using the Qubit RNA BR assay kit (Life Technologies, Carlsbad, CA) and reverse transcribed to cDNA using the High Capacity cDNA reverse transcription kit (Applied Biosystems, Foster, CA), according to manufacturers’ specifications. Gene expression was quantified through the use of real time-PCR (One step Plus, Applied Biosystems); a list of the genes studied can be found in Table 6-1. The values were transformed to ΔCq values using the NormqPCR package (1.7.1) in R (3.3.2).

Table 6-1: Panel of genes assessed. List includes catabolic, inhibitors of catabolism, the HIF 1α transcription factor, and the notochordal marker Brachyury. 18s and RPL4 are used as the reference genes.

<table>
<thead>
<tr>
<th>Gene</th>
<th>Forward Sequence</th>
<th>Reverse Sequence</th>
<th>Amplicon length</th>
<th>Target Gene</th>
</tr>
</thead>
<tbody>
<tr>
<td>18s</td>
<td>CGGCTACCATCCAAGGA</td>
<td>AGCTGGAATACGCGGCT</td>
<td>188</td>
<td>NR_045261.1</td>
</tr>
<tr>
<td>RPL4</td>
<td>ATACAGACCTTACGCAATTTTA</td>
<td>AACTCTCGTGGTCGCTG</td>
<td>71</td>
<td>DQ45176</td>
</tr>
<tr>
<td>Brachyury</td>
<td>AAGTACGTGAAAGGAGGCTG</td>
<td>CACGATGTCATCCGAGGCT</td>
<td>213</td>
<td>XM_001926144.4</td>
</tr>
<tr>
<td>HIF-1α</td>
<td>TTAGACCTAATAGCCGAGGAGGA</td>
<td>AAATAGGGCGAGCCGAGG</td>
<td>310</td>
<td>NM_00123124.1</td>
</tr>
<tr>
<td>ADAMTS4</td>
<td>TCCTGCAACACCCCAAGCTG</td>
<td>CTGAGAGATTCGCTGCGGT</td>
<td>93</td>
<td>XM_003480144.4</td>
</tr>
<tr>
<td>ADAMTS5</td>
<td>AAATGCGATGAGGATGGGA</td>
<td>TAAGGCAAAAATGCAAGG GC</td>
<td>197</td>
<td>XM_0021070864.1</td>
</tr>
<tr>
<td>MMP 3</td>
<td>TTCTGATGATGGTTTACTTCGAC</td>
<td>TTGCACTACTCAGATGAGGCTCATT</td>
<td>78</td>
<td>NM_001166308.1</td>
</tr>
<tr>
<td>MMP 13</td>
<td>CATGAGTTTGCCCATCTCCT</td>
<td>GTGCCTTTTCACAGTGAG</td>
<td>90</td>
<td>XM_003129808.3</td>
</tr>
<tr>
<td>TIMP 1</td>
<td>ATTGGGAGCGCCAGAGCTG</td>
<td>CCAAGGACATTGACCTT</td>
<td>93</td>
<td>NM_213857.1</td>
</tr>
<tr>
<td>TIMP 2</td>
<td>CTACGGCCAACCCGCAACA</td>
<td>AGGAAGGGGCGCGTGTAGATA</td>
<td>101</td>
<td>NM_00145985.1</td>
</tr>
<tr>
<td>TIMP 3</td>
<td>CCGCTGCTTTGCTGGACCT</td>
<td>CGGATGAGGCGTGCTT</td>
<td>102</td>
<td>XM_003126073.4</td>
</tr>
</tbody>
</table>

6.2.6 Statistical Analysis

Five pigs (n=5) were used in the study. A three-way ANOVA was performed on the oxygen tension, loading, and time using base R (v3.6.1), while controlling for individual pig effects for all of the genes and matrix properties studied. Significance was inferred at p<0.05. The R equation for the statistical model can be found in Equation 6-1.
6.3 Results

The bioreactor was successfully able to cycle media and deliver cyclic load to the IVD segments. A brief look at a segment of runtime during a cyclic loading portion is shown in Figure 6-1.

![Figure 6-1: Loading in bioreactor during sample experimental run.](image)
Gross matrix properties, as characterized by water content, the DMMB assay, and the hydroxyproline assay, showed few statistically significant differences except in tissue water content. Controlling for individual pig variability was a consideration in a majority of the assays.

Tissue water content of the NP was significantly affected by the oxygen tension: loading condition interaction (p=0.024) and approached significance for the time: loading condition interaction (p=0.091). AF tissue had a significant difference between discs of different loading conditions (p=0.013) and the influence of oxygen tension approached significance (p=0.075). Pig variability (p=0.012) significantly influenced AF tissue water content. AF tissues tended to increase their water content post-harvest. The NP of the 5% oxygen tension group in the bioreactor trended downward but this relation was not found to be significant. As can be seen in Figure 6-7 below.

Proteoglycan content was not found to be significantly influenced by any factors other than individual variability of the pigs in the AF tissues (p=0.059). There was a tendency for proteoglycan content to decrease in the AF tissue post-harvest in both bioreactor and free-swelling discs. As can be seen in Figure 6-8 below.
Figure 6-7: Water content of NP (A) and (C) and AF (B) and (D) tissue as percent of total mass in the bioreactor (A) and (B) as well as free-swelling discs (C) and (D).

Figure 6-8: Proteoglycan content of NP (A) and (C) and AF (B) and (D) tissue on a dry mass basis in the bioreactor (A) and (B) as well as free-swelling discs (C) and (D). Proteoglycan content was assessed using the DMMB assay.
Hydroxyproline content of the NP tissues showed a significant interaction between oxygen tension and culture time (p=0.043). The effect of pig variability approached significance in both tissues for hydroxyproline content (p=0.088 and p=0.088 for NP and AF tissues respectively). The hydroxyproline content of the NP tissue cultured at 21% oxygen tension had a trend of rapidly increasing post-harvest and then declining, while the 5% oxygen tension groups showed a more gradual increase. For AF tissues the hydroxyproline content rapidly increased post-harvest but then stabilized.

![Figure 6-9: Hydroxyproline content of NP (A) and (C) and AF (B) and (D) tissue on a dry mass basis in the bioreactor (A) and (B) as well as free-swelling discs (C) and (D). Hydroxyproline content was assessed using the Hydroxyproline assay.](image)

Gene expression within the disc showed much more activity as compared to gross matrix changes. Brachyury and HIF-1α, **Figure 6-10**, had no significant differences
between groups, though pig variability was found to influence HIF-1α (p=0.070). Though not statistically significant, Brachyury expression trended downward while HIF-1α expression trended upward post-harvest.

![Gene expression of Brachyury (A) and (B) and HIF-1α (C) and (D) using RT-PCR and the -ΔCq method in both bioreactor (A) and (C) and free-swelling (B) and (D) discs.](image)

There were several significant differences noted in the expression of catabolic genes. MMP 3 was significantly influenced by both oxygen tension (p=0.032) and culture time (p=0.0094). Also, the effects of loading condition were found to significantly interact with oxygen tension (p=0.023) and be influenced by culture time (p=0.083). Pig variability was found to be a significant influence (p=0.040). As can be seen in Figure 6-11, MMP 3 expression rapidly increased post-harvest; only the 21% oxygen tension group in the
bioreactor decreased after an initial spike, while all other groups trended upward. MMP 13 expression was similarly influenced by oxygen tension \( (p=0.045) \) and culture time \( (p=0.0038) \). Additionally, an interaction was found between oxygen tension and loading condition \( (p=0.067) \) for MMP 13 expression. Again MMP 13 expression looked similar to MMP 3 expression with all groups showing increasing expression over time. ADAMTS 4 showed increasing or decreasing trends, but had pig variability play a role in expression \( (p=0.082) \). ADAMTS 5 was found to be significantly influenced by pig variability \( (p=0.022) \) and the effect of culture time approached significance \( (p=0.085) \). The trend of ADAMTS 5 expression was upward for all conditions over time.

The expression of inhibitors to catabolism, in Figure 6-12, was somewhat altered by experimental conditions. TIMP 1 was found to be significantly affected by genetic variability \( (p=0.0037) \) and the effect of culture time approached significance \( (p=0.077) \). Over the culture time, the expression of TIMP 1 trended downward for all groups. A more pronounced trend was seen in TIMP 2 expression. Analysis of TIMP 2 showed that culture time played a significant role \( (p=0.0021) \), along with oxygen tension \( (p=0.0013) \) and pig variability \( (p=0.034) \).
Figure 6-11: Gene expression of catabolic genes ADAMTS 4 (A) and (B), ADAMTS 4 (C) and (D), MMP 3 (E) and (F), MMP 13 (G) and (H), using RT-PCR and the ΔΔCq method in both bioreactor (A), (C), (E), and (G) and free-swelling (B), (D), (F), and (H) discs.
Figure 6-12: Gene expression of catabolic inhibitor genes TIMP 1 (A) and (B), TIMP 2 (C) and (D), and TIMP 3 (E) and (F) using RT-PCR and the \(-\Delta C_T\) method in both bioreactor (A), (C), and (E) and free-swelling (B), (D), and (F) discs.

### 6.4 Discussion

The goal of this study was to analyze the influence of culture oxygen tension on whole IVD segments both inside a custom-built bioreactor and under free-swelling conditions. Several of the genes involved in catabolism were found to be significantly influenced by
oxygen tension and there was a significant interaction with other experimental variables. Additionally, culture time itself was found to be a significant factor in the gene expression of multiple genes. This may be due to a post-harvest adaptation period to the new *in vitro* environment. This adaptation period has been noted in previous chapters, where the cells were still changing their metabolic activity and gene expression significantly over a ten-day period. While many other studies have noted an effect of mechanical loading on IVD cells, no genes or matrix properties showed a significant difference between loaded and free-swelling discs in this study [70, 72, 73, 125, 145, 155, 208, 212]. It was, however, found to be significantly influential in conjunction with other factors in the expression of MMPs.

Contrary to trends seen in the agarose disc studies presented here in Chapters 4 and 5, the TIMP and MMP expression did not coregulate, instead both TIMP 1 and 2 decreased while MMP 3 and 13 were increased over the same interval. Additionally, the ADAMTS 5 gene was also increasing along the same interval. All these factors lead the NP tissue towards catabolism. Though no gross changes in proteoglycan were found, there were changes in hydroxyproline content. These changes, however, showed an initial increase in the hydroxyproline content post-harvest with a declining trend afterword for the 21% oxygen tension. This decreasing trend could potentially be influenced by the rising expression of MMP 3 and MMP 13, which target collagen, and the lack of catabolic protection with the falling expression of TIMPs [40-42, 85, 92, 181, 182, 219, 220]. A study in ovine caudal discs by Junger et al. found a similar trend in ADAMTS 4 expression as the 21% oxygen tension group, which matched that study’s culture conditions, but the
MMP 13 expression was found to have an inverse trend to that found in this study [71]. This may be due to differences in species or cell phenotype as sheep tend to display a more chondrocytic cell phenotype in contrast to the more notochordal cell phenotype of porcine sources [74, 75, 77, 78]. This is in contrast with the 5% oxygen tension group which exhibited increasing hydroxyproline content with increasing culture time, despite an increase in expression of all catabolic genes interrogated and decreases in TIMPs expression. Previous studies report increased expression of MMPs and ADAMTS as well as a decrease in TIMP expression around degenerative changes within IVD tissues [41, 42, 85, 219, 220]. The trends seen in the free-swelling discs are similar to those in the bioreactor. These trends are in agreement with a study by Li et al. in a substance exchange bioreactor on porcine discs [209]. Similar increases in hydroxyproline content and catabolic gene expression changes were observed by Li et al. These discs were also unloaded and mimicked much of the conditions of the free-swelling discs [209].

Though we believe this study provides some valuable insights into the metabolism of the NP of the porcine IVD, there are several shortcomings of the current study which future studies will address. Primarily is the low sample number of the current study. With only five subjects per group, there was insufficient statistical power for the study and further follow up study with additional subjects would improve the results seen here; based on power analysis, a minimum of 8 total subjects would be needed for achieving that statistical power. Additionally, as noted earlier, NP cells adapt many of their metabolic functions over the span of 7-10 days when placed within a new environment [30, 32, 109, 110]. Many of the genes analyzed here exhibited trends between days 0 and 7 while trends
for other timepoints were limited. This would suggest that longer follow-up studies are warranted. Additionally, this would improve the fidelity of the DMMB and hydroxyproline assays, which rely on gross changes to tissue ECM composition to be notable.

We were able to develop a bioreactor system which is capable of simulating physiological loading over extended periods of up to 14 days in culture. The bioreactor was used to investigate the ECM and gene expression of whole IVD segments. No differences were seen between free-swelling and loaded discs, but trends toward degeneration were observed in both oxygen tension conditions and loading conditions. This indicates that culture conditions have not yet been optimized to truly mimic the in vivo environment. With further work and experimentation, bioreactors such as the one used here could potentially be used to maintain tissues and organs for months for use in regenerative treatments. This preliminary work provides valuable insight into which parameters are critical to the functioning of bioreactors and can help illuminate targets for therapy. Moreover, the clear differences in response between oxygen tensions groups indicates that this nutrient is a vital consideration when culturing NP cells. Additionally, disc degeneration could be more closely studied and the prolonged effects of anabolic and catabolic dysregulation monitored to better understand the progression of DDD.
Chapter 7 Conclusions and Recommendations for Future Work

7.1 Summary and Concluding Remarks

Lower back pain is an ailment that will afflict most people at some point in their lifetime with millions becoming chronic suffers. Degeneration of the IVD is believed to be a source of this back pain. Poor nutrition is implicated as a leading factor initiating the degenerative cascade in intervertebral discs, although this link has never been elucidated. Additionally, regenerative medicine techniques will require culturing cells for prolonged periods in vitro and understanding the behavior of intervertebral disc cells under culture conditions will be invaluable knowledge for future treatments.

The major objectives of this dissertation were therefore: (1) better understand degenerative disc disease progression; (2) find metabolic factors influencing the disc and disc degeneration; and (3) search for treatment targets that can treat or reverse tissue degeneration. To achieve these aims, four studies were carried out: (1) prolonged in vitro culture using agarose gels measuring GCR and gene expression in response to oxygen tension (Chapters 3 and 4); (2) prolonged in vitro culture using agarose gels measuring GCR and gene expression in response to glucose concentration (Chapters 3 and 4); (3) in vitro culture using agarose gels to better model GCR and use gene expression to find treatment targets in response to oxygen tension (Chapter 5); and (4) development of a whole IVD bioreactor to simulate physiological stresses while allowing prolonged culture...
while applying altered oxygen tension to better understand ideal culture conditions and treatment targets (Chapter 6). The most important finding from these studies are summarized below.

7.1.1 Effects of Oxygen Tension and Glucose Concentration on NP cells

Oxygen tension was found to have an effect on notochordal phenotype stability as well as several genes involved in anabolism and catabolism. Lower oxygen tension resulted in a more favorable phenotype. Glucose concentration was not found to significantly influence the behavior of NP cells. Glucose consumption rates were also significantly affected by changes in oxygen tension, but only after several days of culture.

The most common reference genes used may not be the most stable and factors like glucose concentration and oxygen tension can alter the stability of selected reference genes.

The glucose consumption rate model was refined to a kinetic model, but due to the limited timeframe no differences in expression were seen. The HIF-1α transcription factor was found to be a likely candidate strongly involved in the regulation of many genes within the notochordal NP.

7.1.2 Development of IVD Bioreactor

We developed a custom IVD bioreactor for whole organ culture of porcine IVDs. The reactor was able to maintain IVDs for 14 days, while applying physiological loading. Discs were then compared to free-swelling discs and evaluated for gene expression and tissue composition.
Some differences were seen in gross hydroxyproline content based on oxygen tension and time, but the other tissue properties were not significantly affected. Gene expression showed trends toward catabolism in both loaded and free-swelling discs. Oxygen tension was found to significantly affect gene expression.

7.2 Recommendations for Future Work

The goal of this research was to provide strategies to combat lower back pain by targeting degenerative disc disease. The information contained within this dissertation provides valuable insights in how notochordal cells respond to environmental factors. There is still, however, much not yet known within the field. Some recommendations for future work are provided to complement the results found here and continue the investigation into the pathophysiology of DDD and potential treatments.

Many genes were investigated over the course of this dissertation, but advances in DNA microarrays allows for the analysis of thousands of genes simultaneously, though at lower fidelity. This would allow us to scrutinize the gene network to identify more candidate genes of interest before narrowing them down using techniques like RT-PCR.

Though the studies presented here lasted longer than many in the literature, the NP is a slow growing tissue, and longer timeframes are needed to accurately understand the cellular behavior. Thus, longer studies both in scaffolds and in bioreactors would be useful to better characterize the NP cell response over therapeutically useful timescales.

Additionally, though the optimal reference genes for porcine NP cells were examined here, no such examination exists for human NP cells and would be a useful
analysis. The reference gene is the ruler against which gene expression is measured, so the more accurate the rule the better the results.

The studies recommended here would greatly enhance the knowledge of the field and further the goals of our research: elucidation of the pathophysiology of lower back pain and preparing for biologically based treatments for the care of degenerative disc disease. These studies will provide new insight into the metabolic and regulatory pathways of IVD cells, which could help identify targets for new treatments and optimize in vitro growth capabilities for cell therapies.
Works Cited


144. Hutton WC, Elmer Wa, Bryce LM, Kozlowska EE, Boden SD, Kozlowski M: Do the intervertebral disc cells respond to different levels of hydrostatic pressure? Clinical biomechanics (Bristol, Avon) 2001, 16(9):728-734.


161. Tommerup N, Leffers H: \textit{Assignment of the human genes encoding 14-3-3 eta (ywhah) to 22q12, 14-3-3 zeta (ywhaz) to 2p25.1–p25.2, and 14-3-3 beta (ywhab) to 20q13.1 by in situ hybridization}. \textit{Genomics} 1996, 33(1):149-150.


182. Xiong C, Zhan X, Xiao Z: **Transcript levels of major mmps and adamts-4 in relation to the clinicopathological profile of patients with tuberculous intervertebral discs and healthy controls.** *Clin Biochem* 2013, 46(7-8):603-611.


188. Altman RB, Raychaudhuri S: **Whole-genome expression analysis: Challenges beyond clustering.** *Curr Opin Struct Biol* 2001, **11**.


131


Appendix A: Technical Drawings

Mechanical Drawings

Appendix A-1: Coupling between linear actuator and load cell
Appendix A- 2: Linking rod between the load cell and porous plate.

Appendix A- 3: Top porous plate.
Appendix A- 4: Bottom porous plate

Appendix A- 5: Spacer for bottom porous plate.
Appendix A- 6: Base plate of bioreactor unit.

Appendix A- 7: Main spine strut of bioreactor unit.
Appendix A-8: Top bracket which hold the bioreactor.

Appendix A-9: Pump and reservoir holder for bioreactor units.
Appendix A-10: Cap of bioreactor vessel.

Appendix A-11: Bioreactor holding plate. Holds two reactor units and the liquid handling system.
Appendix A-12: Holding bracket for the reactor vessel.

Electrical Schematics
Appendix A-13: Schematic for the sensor board.

Appendix A-14: Schematic of the motor driver shield. 1st page of 2.
Appendix A-15: Schematic of motor driver shield. 2nd page of 2.
Appendix A-16: Electrical isolation board.
Appendix B: Source Code

This section contains all source code developed for the bioreactor project.

Main Program

BioreactorController.ino

/*

* Bioreactor Controller

*

* This is the main program to run and log data from a IVD bioreactor.

*

*

* Author: Lukas Jaworski

* Principal Investigator: Alicia Jackson

*

* Version: 1.2.0

*

* University of Miami

* 2018-2019
#include "Arduino.h"
#include <ReactorControl.h>
#include "ReactorConstants.h"
#include <avr/wdt.h>
#include <Wire.h>
#include <SPI.h>
motorControl motorController;

dataHandler dataProcessing;

manualControl manualOverride;

ISR(WDT_vect){
    motorControl::stopLinearActuators();
    dataHandler::writeError(10, 0);
}

ISR(WDT_vect){
    motorControl::stopLinearActuators();
    dataHandler::writeError(10, 0);
}

 ISR(WDT_vect){
    motorControl::stopLinearActuators();
    dataHandler::writeError(10, 0);
}

ISR(WDT_vect){
    motorControl::stopLinearActuators();
    dataHandler::writeError(10, 0);
}

 ISR(WDT_vect){
    motorControl::stopLinearActuators();
    dataHandler::writeError(10, 0);
}

ISR(WDT_vect){
    motorControl::stopLinearActuators();
    dataHandler::writeError(10, 0);
}

ISR(WDT_vect){
    motorControl::stopLinearActuators();
    dataHandler::writeError(10, 0);
}

ISR(WDT_vect){
    motorControl::stopLinearActuators();
    dataHandler::writeError(10, 0);
}
void setup(){

    motorControl::setup();

    motorControl::watchdogSetup();

    Wire.begin();

    // For debugging and monitoring

    // Serial.begin(115200);

    // Switch control pin 8 to ON to update the RTC stored timestamp to the
    // begining of the experiment. make sure to switch it to OFF after the
    // reactors fully boot up.

    if (1 & ~PINC>>7){

        RTC.set(EXPERIMENT_START_TIME);

    }

    // Serial.println("Started Program");

    wdt_reset();
dataProcessing.bootLoadCells();

// Serial.println("booted load cells");

dataProcessing.startSD(53);

dataProcessing.loadProfile();

wdt_reset();

dataProcessing.intializeDatafile();

wdt_reset();

// Serial.println("Beginning Experiment");

}

void loop(){

dataProcessing.updateSetpoints();
dataProcessing.updateLoads();

motorController.adjustLinearActuator(dataProcessing);

wdt_reset();

motorController.removeReactor(dataProcessing);

    // Display the values in real time for monitoring

    // for (uint8_t reactor=0; reactor<4; reactor++){

    // // Check to see if the reactor is still active

    // if ((1 & (~PINC >> reactor))){

    // // manualOverride.displayCompVals(reactor, dataProcessing);

    // } //}

    // }

    // }

wdt_reset();

}

**Bioreactor Library**

*Header files*

ReactorConstants.h
#ifndef ReactorConstants_h

#define ReactorConstants_h

#define EXPERIMENT_START_TIME 1563826568

#define UPDATE_RTC false

#define START_IN_HIGH_MODE false

#define P_VAL_FWD 50000

#define I_VAL_FWD 5000000

#define P_VAL_REV 500000

#define I_VAL_REV 50000000

#define R0_HIGH_MODE_ON_TIME 28800

#define R0_HIGH_MODE_FREQUENCY 2000

#define R0_HIGH_MODE_MAX_LOAD 8200000

#define R0_HIGH_MODE_MIN_LOAD 6821739

#define R0_LOW_MODE_FREQUENCY 2000

#define R0_LOW_MODE_MAX_LOAD 5800210

#define R0_LOW_MODE_MIN_LOAD 5800210
#define R0_ERROR_BOUNDS 60000

#define R0_LOAD_CELL_OFFSET 100000

#define R1_HIGH_MODE_ON_TIME 22800

#define R1_HIGH_MODE_FREQUENCY 2000

#define R1_HIGH_MODE_MAX_LOAD 2968204

#define R1_HIGH_MODE_MIN_LOAD 1033168

#define R1_LOW_MODE_FREQUENCY 2000

#define R1_LOW_MODE_MAX_LOAD 2968204

#define R1_LOW_MODE_MIN_LOAD 1033168

#define R1_ERROR_BOUNDS 60000

#define R1_LOAD_CELL_OFFSET 100000

#define R2_HIGH_MODE_ON_TIME 28800

#define R2_HIGH_MODE_FREQUENCY 2000

#define R2_HIGH_MODE_MAX_LOAD 8200000

#define R2_HIGH_MODE_MIN_LOAD 6944856
#define R2_LOW_MODE_FREQUENCY 1000

#define R2_LOW_MODE_MAX_LOAD 5675923

#define R2_LOW_MODE_MIN_LOAD 5675923

#define R2_ERROR_BOUNDS 60000

#define R2_LOAD_CELL_OFFSET 100000

#define R3_HIGH_MODE_ON_TIME 22800

#define R3_HIGH_MODE_FREQUENCY 2000

#define R3_HIGH_MODE_MAX_LOAD 7396535

#define R3_HIGH_MODE_MIN_LOAD 5795344

#define R3_LOW_MODE_FREQUENCY 1000

#define R3_LOW_MODE_MAX_LOAD 6595940

#define R3_LOW_MODE_MIN_LOAD 5795344

#define R3_ERROR_BOUNDS 60000

#define R3_LOAD_CELL_OFFSET 100000

#endif
ReactorControl.h

/*

Library containing several classes used in the control and configuration of
an IVD bioreactor. The microprocessor system used is the Arduino MEGA 2560
with several optimizations included in the library specific to the ATmega2560
chip and will NOT work with other microprocessors without serious rework of
the code.

The TCA9544 and NAU7802 libraries were written specifically for this project,
but these libraries should be written for general use and can be used anywhere.
All other libraries were pulled from online sources like GitHub and tend to be
part of the Arduino ecosystem.

Authors: Lukas Jaworski, Anthony Wolliston

Principal Investigator: Alicia Jackson

Version: 1.0.0
Required Hardware:

- Arduino MEGA 2560
- controller board
- isolation board
- sensor board
#include "Arduino.h"

#include "ReactorConstants.h"

#include <avr/wdt.h>

#include <Wire.h>

#include <SPI.h>

#include <SD.h>

#include <Time.h>

#include <TimeLib.h>

#include <DS3232RTC.h>

#include <TCA9544.h>

#include <NAU7802.h>


('').repeat(200)
#define REMOVE_PIN 4

#define MAX_PWM_VAL 700

defined PIDdir {

    FWD = 0,

    REV = 1

} ; 

#define PIDstate {

    turnOFF = 0,

    turnON = 1

} ; 

struct PIDvalues {//This struct holds the control PID variables

    int32_t load;

    int16_t force;

    int32_t setpoint;
PIDstate state;

PIDdir dir;

};

struct ProfileParams {

    int32_t freq;

    int32_t minMax[2];

};

struct Profile {

    ProfileParams mode[2];

    int32_t onTime;

    int32_t error;

};

///////////////////////////////////////////////////////////////////////////
class dataHandler {

public:

dataHandler();

void startSD(uint8_t chipSelect = 53);

void loadProfile();

void intializeDatafile();

void bootLoadCells();

void setLoadCellOffset(uint8_t reactor, uint32_t offset);

void storeData(uint8_t reactor);

// write a specified error to the SD card

static void writeError(uint8_t errorCode, uint8_t reactor);
void updateSetpoints();

void updateLoads(bool storeData = true);

void adjOnTime();

int16_t PIDcalculation(uint8_t reactor, int32_t diff, bool changedDirection);

friend class motorControl;

friend class manualControl;

friend class reactorCalibrator;

protected:

TCA9544 channelSelector;

File dataLogger;

uint8_t SD_CS;

PIDvalues PIDvals[4];

int32_t integralBuffer[4][10];

uint32_t reactorTimings[4];

Profile expProfile[4]; // Holds the frequency and load information for
volatile uint8_t phase; // A variable that describes what loading
// pattern to use (the first nibble) and what
// point in the loading frequency phase the
// experiment is in (the second nibble).

volatile uint8_t updated; // this byte will track when the setpoint is updated and
when the loading is changed

time_t expBeginning;

uint8_t startHigh;

uint8_t goingForward; // A byte to track which direction the linear actuator went in
the last PID step.

uint16_t powerOnTime;

uint16_t deloadForce;

};
public:

    motorControl();

    static void setLinearActuator(int8_t reactor, dataHandler& data); //controls the linear actuator in terms of direction and PWM duty cycle

    static void stopLinearActuators(void);

    static void setPWM(void);

    static void watchdogSetup(void);

    static void setPinDirections(void);

    static void setup(void);

    void adjustLinearActuator(dataHandler& data);

    void removeReactor(dataHandler& data);

protected:

    void deload(uint8_t reactor, dataHandler& data);

    void load(uint8_t reactor, dataHandler& data);

}
class manualControl {

public:

    manualControl();

    void receiveCommand(void);
    void parseCmd(dataHandler& data);
    bool executeCommand(dataHandler& data);
    static void displayPIDvals(int8_t reactor, dataHandler& data);
    static void displayCompVals(int8_t reactor, dataHandler& data);

protected:

const byte numChars = 32;

char receivedChars[32];

bool newData;

int8_t reactorNumber;

int8_t dir;

uint16_t force;

uint32_t motorTime;

bool commandReceived;

int32_t actualWeight;

};

class reactorCalibrator: public manualControl {

public:

reactorCalibrator();

void parseCmd(uint8_t reactor, dataHandler& data);
void executeCommand(dataHandler& data);

void motorCalPrep(uint8_t reactor, dataHandler& data);

void calibrateMotors(uint8_t reactor, uint32_t powerUpInterval[5],
                     uint32_t updateInterval, uint16_t forceInc[5],
                     uint16_t deloadForce[5], dataHandler& data);

};

#endif  // ReactorControl

Source

ReactorControl.cpp

/*
 *
 * Library for controlling an IVD Bioreactor using Linear actuators and an
 *
 * NAU7802 load cell reader and an S-load cell. Motors and load cells need to be
 *
 * calibrated before use.
 *
 *
 * A python script called bioreactor.py was created in order to deal with all load
 *
 * calculations, program file management, data analysis, and other parameters
 */
* dealing with the bioreactor experiments.

* Authors: Lukas Jaworski, Anthony Wolliston

* Principal Investigator: Alicia Jackson

* University of Miami 2018

* v 1.0.0

* /

#include "ReactorControl.h"

ключен "ReactorControl.h"

/////////////////////////////////////////////////////////////////////

// dataHandler //

/////////////////////////////////////////////////////////////////////
dataHandler::dataHandler()
{

/*

* This is the object that will be used to write all critical info and data
* to the SD card.
*/

/*

* Initialize all of the calibrated values for the motors and loadcells.

* These values are contained in the ReactorConstants.h file.

* This file can be modified directly by hand to adjust the bioreactor

* parameters or the bioreactor.py script can be run to generate these values

* using a GUI and optional calibration files [once it is complete, if that ever happens]
*/

expProfile[0] = {{R0_LOW_MODE_FREQUENCY,

{R0_LOW_MODE_MIN_LOAD,R0_LOW_MODE_MAX_LOAD}},

{R0_HIGH_MODE_FREQUENCY,

}
\[
\{R0\_{HIGH\_MODE\_MIN\_LOAD},R0\_{HIGH\_MODE\_MAX\_LOAD}\}\},
\]
\[
\text{R0\_{HIGH\_MODE\_ON\_TIME},R0\_{ERROR\_BOUNDS}}; \\
\text{expProfile[1] = \{\{R1\_{LOW\_MODE\_FREQUENCY},}
\]
\[
\text{\{R1\_{LOW\_MODE\_MIN\_LOAD},R1\_{LOW\_MODE\_MAX\_LOAD}\}\},}
\]
\[
\text{\{R1\_{HIGH\_MODE\_FREQUENCY},}
\]
\[
\text{\{R1\_{HIGH\_MODE\_MIN\_LOAD},R1\_{HIGH\_MODE\_MAX\_LOAD}\}\},}
\]
\[
\text{R1\_{HIGH\_MODE\_ON\_TIME},R1\_{ERROR\_BOUNDS}}; \\
\text{expProfile[2] = \{\{R2\_{LOW\_MODE\_FREQUENCY},}
\]
\[
\text{\{R2\_{LOW\_MODE\_MIN\_LOAD},R2\_{LOW\_MODE\_MAX\_LOAD}\}\},}
\]
\[
\text{\{R2\_{HIGH\_MODE\_FREQUENCY},}
\]
\[
\text{\{R2\_{HIGH\_MODE\_MIN\_LOAD},R2\_{HIGH\_MODE\_MAX\_LOAD}\}\},}
\]
\[
\text{R2\_{HIGH\_MODE\_ON\_TIME},R2\_{ERROR\_BOUNDS}}; \\
\text{expProfile[3] = \{\{R3\_{LOW\_MODE\_FREQUENCY},}
\]
\[
\text{\{R3\_{LOW\_MODE\_MIN\_LOAD},R3\_{LOW\_MODE\_MAX\_LOAD}\}\},}
\]
\[
\text{\{R3\_{HIGH\_MODE\_FREQUENCY},}
\]
\[
\text{\{R3\_{HIGH\_MODE\_MIN\_LOAD},R3\_{HIGH\_MODE\_MAX\_LOAD}\}\},}
\]
\[
\text{R3\_{HIGH\_MODE\_ON\_TIME},R3\_{ERROR\_BOUNDS}}; 
\]
//integralBuffer = {0};

for (int8_t j=0; j<4; j++){
    for (int8_t i=0; i<10; i++){
        integralBuffer[j][i] = 0;
    }
}

void dataHandler::startSD(uint8_t chipSelect){
    //Start the SD card to record the loadcell values and any error messages
    SD_CS = chipSelect;
    while (!SD.begin(SD_CS)){
        Serial.println("Insert SD card!");
    }
}

void dataHandler::loadProfile(){
The function name is a legacy name from when the experimental profile was loaded from a .txt file on the SD card. Now the purpose of this function is to set a few local variables needed for the operation of the bioreactor as well as syncing the Arduino clock with the RTC

```cpp
startHigh = START_IN_HIGH_MODE; //weather the bioreactor starts in high or //low mode operation
if (startHigh) phase = 0b11110000;

expBeginning = EXPERIMENT_START_TIME; //UNIX timestamp of when the experiment //began. Written by a python program

//at experiment initialization (make //sure to run the bioreactor.py script)

// Load the offsets need for each loadcell such that it has a low value at // no load
dataHandler::setLoadCellOffset(0, R0_LOAD_CELL_OFFSET);

dataHandler::setLoadCellOffset(1, R1_LOAD_CELL_OFFSET);

dataHandler::setLoadCellOffset(2, R2_LOAD_CELL_OFFSET);

dataHandler::setLoadCellOffset(3, R3_LOAD_CELL_OFFSET);

//only sync with the clock at a system reset as there was some error with

//the setSyncProvider function occasionally updating to the wrong time.

uint16_t countDown = 0;

do {

countDown++;

setTime(RTC.get());

if (countDown > 1000){

dataHandler::writeError(2,0);

Serial.println("4,2,2,2"); //4 is an error code in the python management script (soon to be added)

Serial.println("Unable to sync with clock");

delay(500); //this will force a WDT reset
}

}
while (timeStatus() != timeSet);

wdt_reset();

void dataHandler::intializeDatafile(void){

/*

This method checks the SD card for data files and creates them if they do not
exist or writes that the bioreactor was reset.

*/

//Since there are 4 potential reactors we need to check all of them

for (uint8_t reactor=0; reactor<4; reactor++){

//check to see which reactors are still 'active'

if (1 & (~PINC >> reactor)){

    channelSelector.changeChannel(reactor);

    String currentLine = "";

    String datafile = "reac" + String(reactor) + ".csv";

}
// checking to see if there already is a log file, and if there is

// record a restart

if (SD.exists(datafile)) {

    currentLine = String(now()) + "," + String(millis()) + ",r,r,r";

    dataLogger = SD.open(datafile, FILE_WRITE);

    if (!dataLogger) {

        Serial.println("DATA FILE NOT INITIALIZED!!!");

    }

    dataLogger.println(currentLine);

} else {

    currentLine = "time,millis,load,setpoint,force";

    dataLogger = SD.open(datafile, FILE_WRITE);

    if (!dataLogger) {

        Serial.println("DATA FILE NOT INITIALIZED!!!");

    }

}
void dataHandler::bootLoadCells(void)
{
/* Boots the all of the bioreactor load cells,

* their settings must still be entered and the first conversion needs to be

* explicitly triggered later

*/

for (uint8_t reactor=0; reactor<4; reactor++)
{
    if (1 & (~PINC >> reactor))
    {
        channelSelector.changeChannel(reactor);

        uint16_t countDown = 0;

        do {

NAU7802::bootCycle(); //this function simply starts the analog
//and digital components of the chip. It
//must still be configured
//(offset, gain, etc.)

countDown++;

if (countDown > 1000){

    dataHandler::writeError(4, reactor);

    Serial.println("4,4,4,4");

    Serial.println("Unable to sync with loadcell");

    delay(500);//this will force a WDT reset

}

} while (!NAU7802::checkPowerUp());

void dataHandler::setLoadCellOffset(uint8_t reactor, uint32_t offset){
// put an offset the NAU7802 doesn't underflow into the max 24bit value

    uint8_t highbyte;
    uint8_t middlebyte;
    uint8_t lowbyte;

    lowbyte = offset & 0xFF;
    middlebyte = (offset >> 8) & 0xFF;
    highbyte = (offset >> 16) & 0xFF;

    channelSelector.changeChannel(reactor);
    NAU7802::setGain(x128);
    NAU7802::setConversionRate(SPS20);
    NAU7802::setVLDO(V45);
    NAU7802::selectAVDDsource(true);
    NAU7802::setRegister(0x05, lowbyte);
    NAU7802::setRegister(0x04, middlebyte);
    NAU7802::setRegister(0x03, highbyte);
    NAU7802::setPGAbypassCAP();
    NAU7802::cycleStart();
wdt_reset();

}

void dataHandler::storeData(uint8_t reactor) {

    /*
    A line with the bioreactor's number, load, and date and time when
    samples were taken, is printed in reac#.csv and then the cursor moves to
    the next line.
    */

    String currentLine = "";

    String datafile = "reac" + String(reactor) + ".csv";

    currentLine = String(now()) + "," + String(millis()) + "," +
                  String(PIDvals[reactor].load) + "," +
                  String(PIDvals[reactor].setpoint) + "," +
                  String(PIDvals[reactor].force);

    dataLogger = SD.open(datafile, FILE_WRITE);

    dataLogger.println(currentLine);
dataLogger.close();

void dataHandler::writeError(uint8_t errorCode, uint8_t reactor){

    /*
    Potential Errors:
    unable to load profile      : 1
    unable to sync with RTC     : 2
    now() < expBeginning        : 3 || This means that there is something wrong with the RTC
    Unable to sync loadcell     : 4
    general WDT reset           : 10
    */

    File errorWriter;

    String currentLine = "";

    String datafile = String("reac") + String(reactor) + String(".csv");

    currentLine = String(now()) + "," + String(millis()) + ",e,e," + String(errorCode);
errorWriter = SD.open(datafile, FILE_WRITE);

errorWriter.println(currentLine);

errorWriter.close();

void dataHandler::updateSetpoints(){

    static uint32_t minMaxTick[4] = {0,0,0,0};

    uint32_t currentMillis = millis();

    //cycle through all potential reactors

    for (uint8_t reactor=0;reactor<4;reactor++){

        // Check to see if the reactor is still 'active'

        if (((1 & (~PINC >> reactor)))){

            // Check to see if switch #7 is flipped and if so change the mode of the reactor

            if (((1 & (~PINC >> 6))){

                phase = phase | 0b11110000;

            } else {

                phase = phase & 0b00001111;

            }

        }

    }

}
// check to see if it is time to change the frequency phase

if((currentMillis - minMaxTick[reactor]) > reactorTimings[reactor]){

  minMaxTick[reactor] = currentMillis;

  // flip the frequency phase of the current reactor
  phase ^= (1 << reactor);

  // the reactor timings just in case it has changed
  reactorTimings[reactor] = expProfile[reactor]
    .mode[(1 & (phase >> (reactor + 4)))]
    .freq;

  // update the reactor setpoints
  PIDvals[reactor].setpoint = expProfile[reactor]
    .mode[(1 & (phase >> (reactor + 4)))]
    .minMax[(1 & (phase >> reactor))];
}

}
void dataHandler::updateLoads(bool storeData) {

    for (uint8_t reactor=0; reactor<4; reactor++) {

        if (1 & (~PINC >> reactor)) {

            channelSelector.changeChannel(reactor);

            if (NAU7802::checkDataReady()) {

                // Tell the PID that the load value was updated and that it
                // needs to do some work

                updated |= (1 << reactor);

                PIDvals[reactor].load = NAU7802::readLoad();

                if (storeData) {dataHandler::storeData(reactor);}

            }

        }

    }

}
void dataHandler::adjOnTime(void){

    //This function is intended to be called by a timer ISR and update the
    //profile phase of the reactor

    //calculate how long the experiment has been running

    time_t dTime = now() - expBeginning;

    time_t offsetDay = 86400; // The number of seconds in a day

    //check to see if the RTC clock is moving in the correct direction and log
    //and error if it is not

    if (dTime < 0){

        for (uint8_t reactor = 0; reactor < 4; reactor++){

            if (((1 & (~PINC >> reactor))){

                dataHandler::writeError(2, reactor);

            }

        }

    }

    return;
}
// onTime (profile phase is based off of the 24hr day and is the amount of
// seconds which the reactor is to be in 'high' mode. All onTime
// calculations will be done based on this time-frame

while (dTime > offsetDay){

    dTime -= offsetDay;

}

for (uint8_t reactor = 0; reactor < 4; reactor++){

    if ((1 & (~PINC >> reactor))){

        if (startHigh){

            if (dTime > (expProfile[reactor].onTime)){

                phase &= ~(1 << (reactor + 4));

            } else {

                phase |= (1 << (reactor + 4));

            }

        } else {

            phase |= (1 << (reactor + 4));

        }

    } else {

        } else {
if (dT ime > (offsetDay - (expProfile[reactor].onTime))){

    phase |= (1 << (reactor + 4));

} else {

    phase &= ~(1 << (reactor + 4));

}

}

}

}

}

}

}

}

}

}

}

}

}

}

}

int16_t dataHandler::PIDcalculation(uint8_t reactor, int32_t diff, bool changedDirection){

    static int8_t bufferIndex[4] = {0,0,0,0};

    int32_t integralBufferSum;

    diff = abs(diff);

    // Clear the integral term on direction changes

    if (changedDirection){
        }
PIDvals[reactor].force = 0;

for (int8_t i=0; i<10; i++) {
    integralBuffer[reactor][i] = 0;
}

integralBuffer[reactor][bufferIndex[reactor]] = diff;

bufferIndex[reactor] += 1;

if (bufferIndex[reactor] > 9) {
    bufferIndex[reactor] = 0;
}

for (int8_t i=0; i<10; i++) {
    integralBufferSum += integralBuffer[reactor][i];
}

if (PIDvals[reactor].dir == FWD) {
    return PIDvals[reactor].force + int16_t(diff / P_VAL_FWD + integralBufferSum / I_VAL_FWD);
} else {

return PIDvals[reactor].force + int16_t(diff / P_VAL_REV + integralBufferSum / I_VAL_REV);

void motorControl::setLinearActuator(int8_t reactor, dataHandler& data){

    /*
    changes the direction and the PWM duty cycle of each reactor, using direct
    port manipulation for speed
    */

    // First we check if the reactor is either active, or if we are sending a control signal

if ((data.PIDvals[reactor].state && (1 & ~(PINC >> reactor))) || ((PINC >> reactor) & (~PINC >> REMOVE_PIN) & 1)) {

    if (data.PIDvals[reactor].dir == FWD) {

        switch (reactor) {

            case 0:

                PORTB |= (1<<PINB5); // pin 13

                OCR1B = 0; // pin 12

                OCR1C = uint16_t(data.PIDvals[reactor].force); // pin 11

                return;

            case 1:

                PORTE |= (1<<PINE3); // pin 5

                OCR3B = uint16_t(data.PIDvals[reactor].force); // pin 2

                OCR3C = 0; // pin 3

                return;

            case 2:

                PORTH |= (1<<PINH3); // pin 6

                OCR4C = uint16_t(data.PIDvals[reactor].force); // pin 8

                OCR4B = 0; // pin 7
case 3:

    PORTL |= (1<<PINL3);    // pin 46

    OCR5C = 0;               // pin 44

    OCR5B = uint16_t(data.PIDvals[reactor].force); // pin 45

    return;

}

} else if (data.PIDvals[reactor].dir == REV){

switch (reactor){

    case 0:

        PORTB |= (1<<PINB5);    // pin 13

        OCR1B = uint16_t(data.PIDvals[reactor].force); // pin 12

        OCR1C = 0;               // pin 11

        return;

    case 1:

        PORTE |= (1<<PINE3);    // pin 5

        OCR3B = 0;               // pin 2
OCR3C = uint16_t(data.PIDvals[reactor].force); // pin 3

return;

case 2:

PORTH |= (1<<PINH3);                          // pin 6

OCR4C = 0;                                      // pin 8

OCR4B = uint16_t(data.PIDvals[reactor].force);  // pin 7

return;

case 3:

PORTL |= (1<<PINL3);                           // pin 46

OCR5B = 0;                                      // pin 45

OCR5C = uint16_t(data.PIDvals[reactor].force);  // pin 44

return;

}
PORTB &= ~(1<<PINB5);

OCR1B = 0;

OCR1C = 0;

return;

case 1:

PORTE &= ~(1<<PINE3);

OCR3C = 0;

OCR3B = 0;

return;

case 2:

PORTH &= ~(1<<PINH3);

OCR4C = 0;

OCR4B = 0;

return;

case 3:

PORTL &= ~(1<<PINL3);

OCR5C = 0;
void motorControl::stopLinearActuators(void) {

    /*
     * Disables and turns off all linear actuators
     */

    OCR1C = 0;
    OCR1B = 0;
    OCR3C = 0;
    OCR3B = 0;
    OCR4C = 0;
    OCR4B = 0;
    OCR5C = 0;

OCR5B = 0;

PORTB &= ~(1<<PINB5);

PORTE &= ~(1<<PINE3);

PORTH &= ~(1<<PINH3);

PORTL &= ~(1<<PINL3);

}

void motorControl::setPWM(void){

    

    /*

    //setting up phase corrected PWM for timers 1, 3, 4, and 5

    TCCR1A |= (1<<COM1B1) | (1<<COM1C1) | (1<<WGM11) | (1<<WGM10);
    TCCR1A &= (~(1<<COM1A0) | (1<<COM1A1) | (1<<COM1B0) | (1<<COM1C0));
    TCCR1B |= (1<<CS10);
    TCCR1B &= ~(1<<ICNC1) | (1<<ICES1) | (1<<WGM12) | (1<<WGM13) | (1<<CS11) | (1<<CS12));

*/
TCCR3A |= (1<<COM3B1) | (1<<COM3C1) | (1<<WGM31) | (1<<WGM30);
TCCR3A &=(1<<COM3A0) | (1<<COM3A1) | (1<<COM3B0) | (1<<COM3C0);
TCCR3B |= (1<<CS30);
TCCR3B &=(1<<ICNC3) | (1<<ICES3) | (1<<WGM32) | (1<<WGM33) |
(1<<CS31) | (1<<CS32));

TCCR4A |= (1<<COM4B1) | (1<<COM4C1) | (1<<WGM41) | (1<<WGM40);
TCCR4A &=(1<<COM4A0) | (1<<COM4A1) | (1<<COM4B0) | (1<<COM4C0);
TCCR4B |= (1<<CS40);
TCCR4B &=(1<<ICNC4) | (1<<ICES4) | (1<<WGM42) | (1<<WGM43) |
(1<<CS41) | (1<<CS42));

TCCR5A |= (1<<COM5B1) | (1<<COM5C1) | (1<<WGM51) | (1<<WGM50);
TCCR5A &=(1<<COM5A0) | (1<<COM5A1) | (1<<COM5B0) | (1<<COM5C0);
TCCR5B |= (1<<CS50);
TCCR5B &=(1<<ICNC5) | (1<<ICES5) | (1<<WGM52) | (1<<WGM53) |
(1<<CS51) | (1<<CS52));
void motorControl::watchdogSetup(void) {

    // Set-up the watchdog timer, because occasionally voltage spikes knock out
    // the main loop of the program

    cli();

    wdt_reset(); // make sure the watchdog accumulation register is cleared

    WDTCSR |= (1<<WDCE) | (1<<WDE); // activate the watchdog change enable

    // Now enable a 250ms timeout (WDP 0-3) and the operational mode should be
    // interrupt (WDIE) and system reset (WDE)

    WDTCSR = (1<<WDIE) | (1<<WDE) | (0<<WDP3) | (1<<WDP2) | (0<<WDP1) |
             (0<<WDP0);

    sei();

}

void motorControl::setPinDirections(void) {

    /*
Setting all pins which will be used as output (PWM, direction select, chip select)

*/

// pins 22-23

DDRA |= (1<<DDA1) | (1<<DDA0);

// pins 10-13 and 53 (DDRB 0 - also the slave select for SPI)

DDRB |= (1<<DDB7) | (1<<DDB6) | (1<<DDB5) | (1<<DDB4) | (1<<DDB0);

// pins 30-37 all input as I will be using this port to indicate which

// reactors are still hooked up

DDRC = 0;

// I also am turning on the internal pull-ups for PORTC

PORTC = (1 << PINC0) | (1 << PINC1) | (1 << PINC2) | (1 << PINC3) | (1 << PINC4)
| (1 << PINC5) | (1 << PINC6) | (1 << PINC7);

// pin 18

DDRD |= (1<<DDD3);

// pins 0, 1, 5, 2, 3 (DDRE 0, 1, 3, 4, 5 respectively)

DDRE |= (1<<DDE5) | (1<<DDE4) | (1<<DDE3) | (1<<DDE1) | (1<<DDE0);

// pins A0-A7 (logically mapped)
void motorControl::setup(void){

    DDRF |= (1<<DDF7) | (1<<DDF6) | (1<<DDF5) | (1<<DDF4) | (1<<DDF3) |
    (1<<DDF2) | (1<<DDF1) | (1<<DDF0);

    // pin 4

    DDRG |= (1<<DDG5); 

    // pins 17, 16, 6-9 (DDRH 0, 1, 3-6 respectively)

    DDRH |= (1<<DDH6) | (1<<DDH5) | (1<<DDH4) | (1<<DDH3) | (1<<DDH1) |
    (1<<DDH0);

    // pins 14 and 15 (DDRJ 1 and 0 respectively, not logical)

    DDRJ |= (1<<DDJ1) | (1<<DDJ0);

    // pins A9, A10, A12, and A13

    DDRK |= (1<<DDK5) | (1<<DDK4) | (1<<DDK2) | (1<<DDK1);

    // pins 44-46 (DDRL 3-5 -> 46, 45, 44: order reversed)

    DDRL |= (1<<DDL5) | (1<<DDL4) | (1<<DDL3);

}

/*

* Set all of the pin directions and PWM settings
motorControl::setPWM();

motorControl::setPinDirections();

// In case the watchdog timer caused a reset, make sure all the linear
// actuators are turned off

motorControl::stopLinearActuators();

return;
}

void motorControl::adjustLinearActuator(dataHandler& data){

    //static uint32_t motorRunTime[4];

    //uint32_t currentMillis = millis();

    int32_t diff;

    //cycle through all potential reactors

    for (uint8_t reactor=0;reactor<4;reactor++){

        // Check to see if the reactor is still active and if the current load
// has been updated

if ((1 & (data.updated >> reactor) & (~PINC >> reactor))){

    diff = abs(data.PIDvals[reactor].setpoint - data.PIDvals[reactor].load);

    if (diff > data.expProfile[reactor].error){

        if (data.PIDvals[reactor].setpoint > data.PIDvals[reactor].load){

            motorControl::load(reactor, data);

            Serial.println("loading");

        } else {

            motorControl::deload(reactor, data);

            Serial.println("deloading");

        }

    } else {

        data.PIDvals[reactor].force = 0;

    }

} else {

    motorControl::setLinearActuator(reactor, data);

    data.updated &= ~(1 << reactor);

void motorControl::deload(uint8_t reactor, dataHandler& data) {

    int32_t diff = (data.PIDvals[reactor].setpoint - data.PIDvals[reactor].load);
    bool changedDirection = false;

    // indicate if the direction of linear actuator movement changed

    if (data.PIDvals[reactor].dir == FWD) {
        changedDirection = true;
        data.PIDvals[reactor].state = turnOFF;
        motorControl::setLinearActuator(reactor, data);
    }

    data.PIDvals[reactor].state = turnON;
    data.PIDvals[reactor].dir = REV;
    data.PIDvals[reactor].force = data.PIDcalculation(reactor, diff, changedDirection);

    // replace

    if (data.PIDvals[reactor].force > MAX_PWM_VAL) {

data.PIDvals[reactor].force = MAX_PWM_VAL;

} // indicate the direction of linear actuator movement

} // end motorControl::load

void motorControl::load(uint8_t reactor, dataHandler& data){

    int32_t diff = (data.PIDvals[reactor].setpoint - data.PIDvals[reactor].load);

    bool changedDirection = false;

    // indicate if the direction of linear actuator movement changed

    if (data.PIDvals[reactor].dir == REV){

        changedDirection = true;

        data.PIDvals[reactor].state = turnOFF;

        motorControl::setLinearActuator(reactor, data);

    }

    data.PIDvals[reactor].state = turnON;

    data.PIDvals[reactor].dir = FWD;

    data.PIDvals[reactor].force = data.PIDcalculation(reactor, diff, changedDirection); // replace
if (data.PIDvals[reactor].force > MAX_PWM_VAL)
{
    data.PIDvals[reactor].force = MAX_PWM_VAL;
}

void motorControl::removeReactor(dataHandler& data)
{
    for (int8_t reactor=0; reactor < 4; reactor++)
    {
        if (1 & (PINC >> reactor) & (~PINC >> REMOVE_PIN))
        {
            data.PIDvals[reactor].state = turnON;
            data.PIDvals[reactor].dir = REV;
            data.PIDvals[reactor].force = 300;
            motorControl::setLinearActuator(reactor, data);
        }
        else if (1 & (PINC >> reactor) & (PINC >> REMOVE_PIN))
        {
            data.PIDvals[reactor].state = turnOFF;
            motorControl::setLinearActuator(reactor, data);
        }
    }
}
manualControl::manualControl() {

    boolean newData = false;

    int8_t reactorNumber;

    int8_t dir;

    uint16_t force;

    uint32_t motorTime;

    boolean commandReceived = false;

    int32_t actualWeight;
}

```c
void manualControl::receiveCommand() {

  static boolean recvInProgress = false;

  static byte index = 0;

  char startMarker = '[';

  char endMarker = ']';

  char rc;

  while (Serial.available() > 0 && newData == false) {

    rc = Serial.read();

    if (recvInProgress == true) {

      if (rc != endMarker) {

        receivedChars[index] = rc;

        index++;

        if (index >= numChars) {

          index = numChars - 1;

```
else {

    receivedChars[index] = '\0'; // terminate the string

    recvInProgress = false;

    index = 0;

    newData = true;

}
if (newData) {

    // split the data into its parts

    //[reactorNumber,direction,force,motorTime]

    char * strtokIndx; // this is used by strtok() as an index

    strtokIndx = strtok(receivedChars, "," ); // Get the reactor number

    reactorNumber = atoi(strtokIndx); // convert this part to an integer

    strtokIndx = strtok(NULL, "," ); // get the next part, the direction

    data.PIDvals[reactorNumber].dir = atoi(strtokIndx); // convert to an int

    // 0 for fwd

    // 1 for rev

    strtokIndx = strtok(NULL, "," ); // now get how much force to apply

    data.PIDvals[reactorNumber].force = atoi(strtokIndx); // convert to an int

    strtokIndx = strtok(NULL, "," ); // now to get how long the motor will run

    motorTime = atoi(strtokIndx); // also convert this to an int
data.PIDvals[reactorNumber].setpoint = motorTime; // reuse the PIDvals

    // setpoint parameter

    // as the motor time

    // variable.

newData = false; // The data block was all received so we can indicate

    // that to the program

commandReceived = true; // A whole command was now received and is ready to be

    // processed

}

}


bool manualControl::executeCommand(dataHandler& data){ //setting the linear actuator
to the commanded value

    static int32_t timer = 0;

    static boolean carryOutCommand = false;

    int32_t timeDiff = millis()-timer;

    if (timeDiff > motorTime && carryOutCommand){

        data.PIDvals[reactorNumber].state = turnOFF;

    }
motorControl::setLinearActuator(reactorNumber, data);

carryOutCommand = false;

return true;

}

if (commandReceived){

    timer = millis();

    data.PIDvals[reactorNumber].state = turnON;

    motorControl::setLinearActuator(reactorNumber, data);

    carryOutCommand = true;

    commandReceived = false;

    }

return false;

}

void manualControl::displayPIDvals(int8_t reactor, dataHandler& data){

    Serial.print("Reactor: ");

    Serial.print(reactor);

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].state);

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].value);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].setPoint);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].error);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].integral);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].derivative);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].dutyCycle);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].output);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].calcTime);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].controlMode);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].controlValue);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].controlGain);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].controlType);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].controlDelay);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].controlLimit);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].controlOffset);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].controlSample);

    Serial.println(" ");

    Serial.println(" ");

    Serial.print(" ");

    Serial.print(data.PIDvals[reactor].controlResolution);

    Serial.println(" ");

    Serial.println(" ");
Serial.print(" | Load: ");

Serial.print(data.PIDvals[reactor].load);

Serial.print(" | Setpoint: ");

Serial.print(data.PIDvals[reactor].setpoint);

Serial.print(" | Force: ");

Serial.println(data.PIDvals[reactor].force);

}

void manualControl::displayCompVals(int8_t reactor, dataHandler& data){

    Serial.print(reactor);

    Serial.print(",");

    Serial.print(data.PIDvals[reactor].load);

    Serial.print(",");

    Serial.print(data.PIDvals[reactor].setpoint);

    Serial.print(",");

    Serial.println(data.PIDvals[reactor].force);

}
#define LOWER_LOADCELL_BOUND 10000

reactorCalibrator::reactorCalibrator() {
    boolean newData = false;

    int8_t reactorNumber;

    int8_t dir;

    uint16_t force;

    uint32_t motorTime;

    boolean commandReceived = false;

    int32_t actualWeight;
}
void reactorCalibrator::parseCmd(uint8_t reactor, dataHandler& data) {

    /*
    * Take the data input string and parse the values
    */

    if (newData) {
        // split the data into its parts
        actualWeight = atoi(receivedChars);  
        Serial.print("The actual weight is : ");
        Serial.print(actualWeight);
        Serial.println("kg");
        reactorNumber = reactor;
        newData = false;
        commandReceived = true;
    }
}
void reactorCalibrator::executeCommand(dataHandler& data) {

  /*
  * Sets the linear actuator to the commanded value
  */

  static int32_t timer = 0;
  static boolean carryOutCommand = false;

  int32_t timeDiff = millis()-timer;

  if ((timeDiff > 1000) && carryOutCommand) {
    carryOutCommand = false;
  } else if ((timeDiff <= 1000) && carryOutCommand) {

    data.PIDvals[reactorNumber].setpoint = actualWeight;

    data.updateLoads();

    reactorCalibrator::displayCompVals(reactorNumber, data);
  }
}
if (commandReceived) {
    timer = millis();
    carryOutCommand = true;
    commandReceived = false;
}

}

void reactorCalibrator::motorCalPrep(uint8_t reactor, dataHandler& data) {
    data.PIDvals[reactor].force = 250;
    data.PIDvals[reactor].state = turnON;
    data.PIDvals[reactor].dir = REV;
    motorControl::setLinearActuator(reactor, data);
    manualControl::displayCompVals(reactor, data);
    delay(200);
data.PIDvals[reactor].state = turnOFF;

motorControl::setLinearActuator(reactor, data);

wdt_reset();

data.PIDvals[reactor].state = turnON;

data.PIDvals[reactor].dir = FWD;

data.PIDvals[reactor].force = 80;

motorControl::setLinearActuator(reactor, data);

delay(200);

wdt_reset();

delay(200);

wdt_reset();

delay(200);

wdt_reset();

data.PIDvals[reactor].state = turnOFF;

motorControl::setLinearActuator(reactor, data);
}

```cpp
void reactorCalibrator::calibrateMotors(uint8_t reactor, uint32_t powerUpInterval[5],
uint32_t updateInterval, uint16_t forceInc[5], uint16_t deloadForce[5], dataHandler& data) {

    // a function to quickly help test the loading of the bioreactor

    /*
    reactor:            reactor number(0-3 as they are zero indexed)
    powerUpInterval:    how long the motor applies power
    updateInterval:     how often the motor increases the load
    forceInc:           how much to increase the motor output in each iteration
    deloadForce:        the motor output during deload actions
    data.PIDvals:       the PID values of the reactor
    */

    static bool deload = false;
    static uint32_t previousTick = millis();
    static uint8_t calRun = 0;

    data.PIDvals[reactor].setpoint = powerUpInterval[calRun];

    if ((millis() - previousTick) > powerUpInterval[calRun]) {
        data.PIDvals[reactor].state = turnOFF;
    }
```
motorControl::setLinearActuator(reactor, data);

data.updateLoads();

} else { data.updateLoads(false);}

// put a new motor load in every update interval

if (!deload){

data.PIDvals[reactor].dir = FWD;

if (data.PIDvals[reactor].load < LOWER_LOADCELL_BOUND){

data.PIDvals[reactor].state = turnON;

data.PIDvals[reactor].force = 80;

motorControl::setLinearActuator(reactor, data);

previousTick = millis();

manualControl::displayCompVals(reactor, data);

}

if (((millis() - previousTick) > updateInterval)){

data.PIDvals[reactor].force += forceInc[calRun];

data.PIDvals[reactor].state = turnON;

motorControl::setLinearActuator(reactor, data);

previousTick = millis();

manualControl::displayCompVals(reactor, data);

}

//Start a deload cycle once we hit the max motor-controller output

if (((data.PIDvals[reactor].force > 300) || ((data.PIDvals[reactor].load > 7000000) && (data.PIDvals[reactor].load < 8400000)))){

data.PIDvals[reactor].force = deloadForce[calRun];

data.PIDvals[reactor].dir = REV;

data.PIDvals[reactor].state = turnOFF;

deload=true;

}

} else {
data.PIDvals[reactor].dir = REV;

if (((millis() - previousTick) > updateInterval)) {

data.PIDvals[reactor].state = turnON;

motorControl::setLinearActuator(reactor, data);

previousTick = millis();

manualControl::displayCompVals(reactor, data);

}

// Once the deload is done repeat the process so I can get a few run together

if (((data.PIDvals[reactor].load < 4700000) || (data.PIDvals[reactor].load > 8000000))
    && deload) {

    data.PIDvals[reactor].state = turnOFF;

    motorControl::setLinearActuator(reactor, data);

    manualControl::displayPIDvals(reactor, data);

    Serial.println("stopping");

    wdt_reset();
reactorCalibrator::motorCalPrep(reactor, data);

calRun++;

Serial.print("Loading Calibration Run ");

Serial.print(calRun);

Serial.println(" Complete!");

deload = false;

while (calRun > 5){

    wdt_reset();

    Serial.println("All Calibration Runs Complete!");

}

wdt_reset();

}

Load Cell Weight Calibration

weightCalibration.ino

/*

Weight Calibration of Bioreactor Loadcells
This program is intended to be used with the IVD bioreactor to calibrate the loadcells.

The loadcells must be taken off of the reactor arms and the weight calibration plates need to be installed. The user then needs to place weights onto the loadcell and input the total weight. This information is stored in a datafile and then used to transform the loadcell readings to real weights.

Lukas Jaworski

Alicia Jackson

University of Miami

2018

*/

///////////////////////////////////////////////////////////////////////////

// Optional Defines

///////////////////////////////////////////////////////////////////////////
```cpp
#include "Arduino.h"

#include <avr/wdt.h>

#include <Wire.h>

#include <SPI.h>

#include <ReactorControl.h>

#include <Adafruit_MotorShield.h>

#include "utility/Adafruit_MS_PWMServoDriver.h"
```
// Global Variables/Objects

motorControl motorController;

dataHandler dataProcessing;

reactorCalibrator calibrator;

// Functions

// ISRs
// For safety we make sure all of the linear actuators are unpowered before the

// watch dog timer resets the system

ISR(WDT_vect){

  motorControl::stopLinearActuators();

}

void setup(){

  motorControl::setup();

  motorControl::watchdogSetup();

  Wire.begin();

  Serial.begin(115200);

  wdt_reset();
Serial.println("RESETTING...");

dataProcessing.bootLoadCells();

dataProcessing.startSD(53);

// Loading the experimental profile from the SD card

dataProcessing.loadProfile();

Serial.println("Beginning Experiment");

wdt_reset();

dataProcessing.intializeDatafile();

wdt_reset();

void loop(){

    wdt_reset();

    dataProcessing.updateLoads(false);
for (uint8_t reactor=0; reactor<4; reactor++) {

    if (1 & (~PINC >> reactor)) {

        calibrator.displayCompVals(reactor, dataProcessing);

    }

}

Python Interface

bioreactor.py

#!/usr/bin/env python3

import subprocess as sub
import itertools
import sys
import os
import tkinter as tk
import tkinter.filedialog
import tkinter.messagebox as msgbx
import pandas as pd

from sklearn.linear_model import LinearRegression

import time

import matplotlib

matplotlib.use('TkAgg')

from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg

from matplotlib.figure import Figure

import matplotlib.pyplot as plt

import matplotlib.animation as ani

import serial

import serial.tools.list_ports as port_list

from collections import deque

import yaml

__author__ = 'Lukas Jaworski'

__version__ = '0.7.2'
#TODO Add comments you neanderthalic dolt

class DataGraph(tk.Frame):

    ""
    This page displays the bioreactor output in real time for all four reactors
    ""

def __init__(self, parent, controller):
    tk.Frame.__init__(self, parent)
    label = tk.Label(self, text="Real Time Graph")
    label.pack(pady=5, padx=5)
    self.bind("<<ShowPage>>", self.show_page_event)

    return_main = ttk.Button(self, text='Return to main screen',
                              command=lambda: controller.show_page(Uploader))
    return_main.pack(side=tk.BOTTOM)

    self.prep_graph()
canvas = FigureCanvasTkAgg(self.fig, self)

canvas.draw()

canvas.get_tk_widget().pack(side=tk.BOTTOM, fill=tk.BOTH, expand=True)

self.start_graph()

def show_page_event(self, event):

    ""

    When the page is shown the serial connection needs to be reopened so
    that data can be received. The graph animation is also restarted.

    ""

    global _serial_connection

    if not _serial_connection.isOpen():
        try:
            _serial_connection.open()
        except:
            msgbx.showerror(title='Connectivity Problems',
                            message='Serial Port Not Found!')
# Restart the animation for the bioreactor output graph

global real_time_graph

real_time_graph.event_source.start()

def prep_graph(self):
    
    """
    The parameters for the graph are stored here as well as initializing all
    of the various artist objects that will be used in the animation
    function. The 'width' of the graph is stored in the variable datapoints
    and dictates how many of the most recent values are held and displayed
    on the graph.
    """

    # Define how many datapoints to use for the plot, basically the history
    # of the setpoints, load, and force.

    self.datapoints = 100
# use double ended ques to simplify adding and popping data on and off
# the stack

self.reactors = [{'load': deque([0]*self.datapoints, self.datapoints),
                 'setpoint': deque([0]*self.datapoints, self.datapoints),
                 'force': deque([0]*self.datapoints, self.datapoints)}
                 for x in range(4)]

self.fig = Figure(figsize=(16,8), dpi=100)

self.ax0 = self.fig.add_subplot(221)

self.ax0_1 = self.ax0.twinx()

self.ax1 = self.fig.add_subplot(222)

self.ax1_1 = self.ax1.twinx()

self.ax2 = self.fig.add_subplot(223)

self.ax2_1 = self.ax2.twinx()

self.ax3 = self.fig.add_subplot(224)

self.ax3_1 = self.ax3.twinx()
self.ax0.set_title('Reactor 1')
self.ax0.set_ylabel('Load')

# The y axis options for setpoint and current load
self.ax0.set_ylim(-10000,8000000)

# The y axis for the PWM signal
self.ax0_1.set_ylim(-10,1024)
self.ax0_1.set_ylabel('Force', color='green')

self.ax1.set_title('Reactor 2')
self.ax1.set_ylabel('Load')

# The y axis options for setpoint and current load
self.ax0.set_ylim(-10000,8000000)

# The y axis for the PWM signal
self.ax0_1.set_ylim(-10,1024)
self.ax0_1.set_ylabel('Force', color='green')
self.ax2.set_title('Reactor 3')

self.ax2.set_ylabel('Load')

# The y axis options for setpoint and current load

self.ax0.set_ylim(-10000,8000000)

# The y axis for the PWM signal

self.ax0_1.set_ylim(-10,1024)

self.ax0_1.set_ylabel('Force', color='green')

self.ax3.set_title('Reactor 4')

self.ax3.set_ylabel('Load')

self.ax3.set_ylim(-10000,8000000)

# The y axis options for setpoint and current load

self.ax0.set_ylim(-10000,8000000)

# The y axis for the PWM signal

self.ax0_1.set_ylim(-10,1024)

self.ax0_1.set_ylabel('Force', color='green')
self.line0_load, = self.ax0.plot(list(range(0, self.datapoints)),
list(self.reactors[0]['load']),'r-')

self.line0_setpoint, = self.ax0.plot(list(range(0, self.datapoints)),
list(self.reactors[0]['setpoint']),'b-')

self.line0_force, = self.ax0_1.plot(list(range(0, self.datapoints)),
list(self.reactors[0]['force']),'g-')

self.line1_load, = self.ax1.plot(list(range(0, self.datapoints)),
list(self.reactors[1]['load']),'r-')

self.line1_setpoint, = self.ax1.plot(list(range(0, self.datapoints)),
list(self.reactors[1]['setpoint']),'b-')
self.line1_force, = self.ax1_1.plot(list(range(0, self.datapoints)),
    list(self.reactors[1]['force']),
    'g-')

self.line2_load, = self.ax2.plot(list(range(0, self.datapoints)),
    list(self.reactors[2]['load']),
    'r-')

self.line2_setpoint, = self.ax2.plot(list(range(0, self.datapoints)),
    list(self.reactors[2]['setpoint']),
    'b-')

self.line2_force, = self.ax2_1.plot(list(range(0, self.datapoints)),
    list(self.reactors[2]['force']),
    'g-')

self.line3_load, = self.ax3.plot(list(range(0, self.datapoints)),
    list(self.reactors[3]['load']),
    'r-')
self.line3_setpoint, = self.ax3.plot(list(range(0, self.datapoints)),
    list(self.reactors[3]['setpoint']),
    'b-')

self.line3_force, = self.ax3_1.plot(list(range(0, self.datapoints)),
    list(self.reactors[3]['force']),
    'g-')

def start_graph(self):
    
    ""
    a simple function to start the graph and have this be callable from
    tkinters various event handlers.
    ""

    global real_time_graph

    real_time_graph = ani.FuncAnimation(self.fig,
        self.updatePlot,
        interval=1,
def updateData(self):

    ""
    Grabs the data from the serial connection if it is ready. If there are
    no updates then it does nothing.
    ""
    try: # ignore any corrupt data or bypass this step if the buffer is empty

        data = _serial_connection.readline().split(b',')

        self.reactors[int(data[0])]['load'].append(int(data[1]))

        self.reactors[int(data[0])]['setpoint'].append(int(data[2]))

        self.reactors[int(data[0])]['force'].append(int(data[3]))

    except:

        pass

def updatePlot(self, i):
self.updateData()

self.line0_load.set_ydata(list(self.reactors[0]['load']))
self.line0_setpoint.set_ydata(list(self.reactors[0]['setpoint']))
self.line0_force.set_ydata(list(self.reactors[0]['force']))

self.line1_load.set_ydata(list(self.reactors[1]['load']))
self.line1_setpoint.set_ydata(list(self.reactors[1]['setpoint']))
self.line1_force.set_ydata(list(self.reactors[1]['force']))

self.line2_load.set_ydata(list(self.reactors[2]['load']))
self.line2_setpoint.set_ydata(list(self.reactors[2]['setpoint']))
self.line2_force.set_ydata(list(self.reactors[2]['force']))

self.line3_load.set_ydata(list(self.reactors[3]['load']))
self.line3_setpoint.set_ydata(list(self.reactors[3]['setpoint']))
self.line3_force.set_ydata(list(self.reactors[3]['force']))
return self.line0_load, self.line0_setpoint, self.line0_force, \
    self.line1_load, self.line1_setpoint, self.line1_force, \
    self.line2_load, self.line2_setpoint, self.line2_force, \
    self.line3_load, self.line3_setpoint, self.line3_force

class MotorCalibration(tk.Frame):
    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)

        self.label = tk.Label(self, text="Motor Calibration")
        self.label.grid(row=0, column=0, columnspan=2, sticky='ew')

        self.bind("<<ShowPage>>", self.show_page_event)

        self.return_main = ttk.Button(self, text='Return to main screen',
                                     command=lambda: controller.show_page(Uploader))

        self.return_main.grid(row=2, column=2)
```python
self.control_label = ttk.LabelFrame(self, text='Calibration Parameters')
self.control_label.grid(row=1, column=2)

self.reactor_selection = tk.IntVar()

self.combobox_label = ttk.LabelFrame(self.control_label, text='Reactor')
self.combobox_label.grid(row=0, column=0, sticky='ew')

self.reactor_combobox = ttk.Combobox(self.combobox_label, textvariable=self.reactor_selection)
self.reactor_combobox['values'] = list(range(1,5))
self.reactor_combobox['state'] = 'readonly'
self.reactor_combobox.current(0)
self.reactor_combobox.pack(side='right')

self.calibration_filename = tk.StringVar()

self.filename_label = ttk.LabelFrame(self.control_label, text='Filename')
self.filename_label.grid(row=1, column=0, sticky='ew')

self.cal_filename_box = ttk.Entry(self.filename_label, textvariable=self.calibration_filename)
```
textvariable=self.calibration_filename)

self.cal_filename_box.pack()

self.cal_direction = tk.StringVar()

self.direction_label = ttk.LabelFrame(self.control_label,
                                      text='Direction [0]-Forward [1]-Reverse')

self.direction_label.grid(row=2, column=0, sticky='ew')

self.direction_box = ttk.Entry(self.direction_label,
                                textvariable=self.cal_direction)

self.direction_box.pack()

self.force = tk.StringVar()

self.force_label = ttk.LabelFrame(self.control_label,
                                  text='Force [PWM signal]')

self.force_label.grid(row=3, column=0, sticky='ew')

self.force_box = ttk.Entry(self.force_label,
                           textvariable=self.force)
self.force_box.pack()

self.motor_time = tk.StringVar()

self.time_label = ttk.LabelFrame(self.control_label, text='Motor Time (ms)')

self.time_label.grid(row=4, column=0, sticky='ew')

self.time_box = ttk.Entry(self.time_label, textvariable=self.motor_time)

self.time_box.pack()

self.send_command = ttk.Button(self.control_label,

    text='Send Command',

    command=self.send_command)

self.send_command.grid(row=5, column=0, sticky='ew')

self.prep_graph()

self.canvas = FigureCanvasTkAgg(self.fig, self)

self.canvas.draw()

self.canvas.get_tk_widget().grid(row=1, column=0, rowspan=2, columnspan=2,
sticky='nesw')
self.start_graph()

def show_page_event(self, event):

    global _serial_connection

    if not _serial_connection.isOpen():

        try:

            _serial_connection.open()

        except:

            msgbx.showerror(title='Connectivity Problems',

                            message='Serial Port\n{}\nNot Found!'.format(_cli_args.port))

    global motor_calibration_graph

    motor_calibration_graph.event_source.start()


def send_command(self):

    global _serial_connection

    if not _serial_connection.isOpen():

        try:
_serial_connection.open()

except:

print('error')

serial_msg = '[' + ','.join([str(self.reactor_selection.get()-1),
                            self.cal_direction.get(),
                            self.force.get(),
                            self.motor_time.get()]) + ']

_serial_connection.write(bytearray(serial_msg, 'ascii'))

def prep_graph(self):

    self.datapoints = 100

    self.reactor = {'load': deque([0]*self.datapoints, self.datapoints),
                    'setpoint': deque([0]*self.datapoints, self.datapoints),
                    'force': deque([0]*self.datapoints, self.datapoints)}

    self.fig = Figure(dpi=100)
self.ax = self.fig.add_subplot(111)

self.ax.set_ylim(10000,16000000)

self.ax.set_title('Current Load Cell Reading')

self.line_load, = self.ax.plot(list(range(0,self.datapoints)),list(self.reactor['load']),'r-')

self.line_setpoint, =

self.ax.plot(list(range(0,self.datapoints)),list(self.reactor['setpoint']),'b-')

def start_graph(self):

    global motor_calibration_graph

    motor_calibration_graph = ani.FuncAnimation(self.fig, self.updatePlot, interval=10,
        blit=True)

def updateData(self):

    global _cli_args

    data_updated = False

    try: #ignore any corrupt data or bypass this step if the buffer is empty

        data = _serial_connection.readline().split(b',')


self.reactor['load'].append(int(data[1]))

self.reactor['setpoint'].append(int(data[2]))

self.reactor['force'].append(int(data[3]))

data[0] = bytearray(self.cal_direction.get(), 'ascii')

file_data = b','.join(data).decode('utf-8')

data_updated = True

except:

data_updated = False

if data_updated:

    cal_file_directory = _cli_args.directory + 'data/calibration/' + 
    self.calibration_filename.get()

    with open(cal_file_directory, "a") as calibration_file:

        print(file_data, file=calibration_file)

    data_updated = False

def updatePlot(self, i):

    self.updateData()
self.line_load.set_ydata(list(self.reactor['load']))

self.line_setpoint.set_ydata(list(self.reactor['setpoint']))

return self.line_load, self.line_setpoint

class WeightCalibration(tk.Frame):

    def __init__(self, parent, controller):

        tk.Frame.__init__(self, parent)

        self.label = tk.Label(self, text="Real Time Graph")

        self.label.grid(row=0, column=0, columnspan=True)

        self.bind("<<ShowPage>>", self.show_page_event)

        self.return_main = ttk.Button(self, text='Return to main screen', command=lambda:
            controller.show_page(Uploader))

        self.return_main.grid(row=2, column=0, columnspan=True)

        self.control_label = ttk.LabelFrame(self, text='Calibration Parameters')

        self.control_label.grid(row=1, column=3)
self.reactor_selection = tk.IntVar()

self.combobox_label = ttk.LabelFrame(self.control_label, text='Reactor')

self.combobox_label.grid(row=1, column=0, sticky='ew')

self.reactor_combobox = ttk.Combobox(self.combobox_label, textvariable=self.reactor_selection)

self.reactor_combobox['values'] = list(range(1,5))

self.reactor_combobox['state'] = 'readonly'

self.reactor_combobox.current(0)

self.reactor_combobox.pack(fill=tk.BOTH, expand=True)

self.calibration_filename = tk.StringVar()

self.filename_label = ttk.LabelFrame(self.control_label, text='Filename')

self.filename_label.grid(row=2, column=0, sticky='ew')

self.cal_filename_box = ttk.Entry(self.filename_label, textvariable=self.calibration_filename)

self.cal_filename_box.pack(fill=tk.BOTH, expand=True)

self.actual_weight = tk.StringVar()
self.weight_label = ttk.LabelFrame(self.control_label, text='Actual Weight (kgs)')

self.weight_label.grid(row=3, column=0, sticky='ew')

self.weight_box = ttk.Entry(self.weight_label, textvariable=self.actual_weight)

self.weight_box.pack(fill=tk.BOTH, expand=True)

self.send_command = ttk.Button(self.control_label, text='Record Data', command=self.send_command)

self.send_command.grid(row=5, column=0, columnspan=True)

self.prep_graph()

self.canvas = FigureCanvasTkAgg(self.fig, self)

self.canvas.draw()

self.canvas.get_tk_widget().grid(row=1, column=0, columnspan=2, sticky='nesw')

self.start_graph()

def show_page_event(self, event):
    global _serial_connection

    if not _serial_connection.isOpen():
try:

    _serial_connection.open()

e except:

    msgbx.showerror(title='Connectivity Problems', message='Serial Port\n{}\nNot Found!'.format(_cli_args.port))

    global weight_calibration_graph

    weight_calibration_graph.event_source.start()

def send_command(self):

    load_cell_readings = self.reactor['load']

    cal_file_directory = _cli_args.directory + 'data/calibration/' +
    self.calibration_filename.get()

    with open(cal_file_directory, "a") as calibration_file:

        for value in load_cell_readings:

            print(self.actual_weight.get() + ',' + str(value), file=calibration_file)


def prep_graph(self):

    self.datapoints = 100
self.reactor = {'load': deque([0]*self.datapoints, self.datapoints),

'setpoint': deque([0]*self.datapoints, self.datapoints),

'force': deque([0]*self.datapoints, self.datapoints)}

self.fig = Figure(figsize=(10,6), dpi=100)

self.ax = self.fig.add_subplot(111)

self.ax.set_ylim(10000,8000000)

self.ax.set_title('Loadcell Reading')

self.line_load, = self.ax.plot(list(range(0,self.datapoints)),list(self.reactor['load']),'r-')

self.line_setpoint, =

self.ax.plot(list(range(0,self.datapoints)),list(self.reactor['setpoint']),'b-')

def start_graph(self):

    global weight_calibration_graph

    weight_calibration_graph = ani.FuncAnimation(self.fig, self.updatePlot, interval=1,
blit=True)

def updateData(self):

    def start_graph(self):

    global weight_calibration_graph

    weight_calibration_graph = ani.FuncAnimation(self.fig, self.updatePlot, interval=1,
blit=True)
global _cli_args

# if _serial_connection.inWaiting():

    # data = _serial_connection.readline().split(b',')

    # self.reactor['load'].append(int(data[1]))

    # self.reactor['setpoint'].append(int(data[2]))

    # self.reactor['force'].append(int(data[3]))

try: #ignore any corrupt data or bypass this step if the buffer is empty

    data = _serial_connection.readline().split(b',')

    self.reactor['load'].append(int(data[1]))

    self.reactor['setpoint'].append(int(data[2]))

    self.reactor['force'].append(int(data[3]))

except:

    pass


def updatePlot(self, i):

    self.updateData()
self.line_load.set_ydata(list(self.reactor['load']))
self.line_setpoint.set_ydata(list(self.reactor['setpoint']))

return self.line_load, self.line_setpoint

class Uploader(tk.Frame):

def __init__(self, parent, controller):

tk.Frame.__init__(self, parent)

global _cli_args

self.controller = controller

self.bind('<<ShowPage>>', self.show_page_event)

self.prog_selection = tk.StringVar()

self.prog_label = ttk.LabelFrame(self, text='Program Selection')

self.prog_label.grid(column=0, row=0)

self.prog_combobox = ttk.Combobox(self.prog_label, textvariable=self.prog_selection)
self.prog_combobox['values'] = ('Main Program', 'Motor Calibration', 'Weight Calibration')

self.prog_combobox['state'] = 'readonly'

self.prog_combobox.current(0)

self.prog_combobox.grid(column=0, row=0, sticky='w')

self.main_prog_update_start_time = tk.IntVar()

self.main_prog_checkbox = ttk.Checkbutton(self.prog_label, text='Update Experiment start time?',
                                          variable=self.main_prog_update_start_time)

self.main_prog_checkbox.grid(column=1, row=0, sticky='w', rowspan=2)

self.main_prog_update_rtc = tk.IntVar()

self.update_rtc_checkbox = ttk.Checkbutton(self.prog_label, text='Update RTC?',
                                           variable=self.main_prog_update_rtc)

self.update_rtc_checkbox.grid(column=1, row=2, sticky='w', rowspan=2)

self.port_selection = tk.StringVar()
self.port_label = ttk.LabelFrame(self.prog_label, text='port')

self.port_label.grid(column=0, row=1, sticky='we')

self.port_combobox = ttk.Combobox(self.port_label, textvariable=self.port_selection)

textvariable= self.port_selection)

self.port_combobox['values'] = [port.device for port in port_list.comports()]

self.port_combobox['state'] = 'readonly'

try:

    self.port_combobox.current(0)

except Exception as err_val:

    print(err_val)

_cli_args.port = self.port_selection.get()

self.port_combobox.bind('<<ComboboxSelected>>', self.update_port)

self.port_combobox.pack(expand=True)

self.upload_button = ttk.Button(self.prog_label, text='Upload', command=self.upload_code)

self.upload_button.grid(column=3, row=0, sticky='ew')
self.graph_button = ttk.Button(self.prog_label, text='Open Graph',
command=self.show_graph_page)

self.graph_button.grid(column=3, row=1, sticky='ew')

self.calibration_label = ttk.LabelFrame(self, text='Select Calibration Files')

self.calibration_label.grid(row=0, column=3, rowspan=10, padx=10, sticky='e')

self.reactor_cal_files = [tk.StringVar() for x in range(4)]

self.reactor1_cal = ttk.Entry(self.calibration_label,
textvariable=self.reactor_cal_files[0])

self.reactor1_cal.grid(row=0, column=1)

self.reactor2_cal = ttk.Entry(self.calibration_label,
textvariable=self.reactor_cal_files[1])

self.reactor2_cal.grid(row=1, column=1)

self.reactor3_cal = ttk.Entry(self.calibration_label,
textvariable=self.reactor_cal_files[2])

self.reactor3_cal.grid(row=2, column=1)

self.reactor4_cal = ttk.Entry(self.calibration_label,
textvariable=self.reactor_cal_files[3])

self.reactor4_cal.grid(row=3, column=1)
self.reactor1_wt_label = ttk.Label(self.calibration_label, text='Reactor 1 calibration files (.cal):')

self.reactor1_wt_label.grid(row=0, column=0)

self.reactor2_wt_label = ttk.Label(self.calibration_label, text='Reactor 2 calibration files (.cal):')

self.reactor2_wt_label.grid(row=1, column=0)

self.reactor3_wt_label = ttk.Label(self.calibration_label, text='Reactor 3 calibration files (.cal):')

self.reactor3_wt_label.grid(row=2, column=0)

self.reactor4_wt_label = ttk.Label(self.calibration_label, text='Reactor 4 calibration files (.cal):')

self.reactor4_wt_label.grid(row=3, column=0)

self.reactor1_wt_button = ttk.Button(self.calibration_label, text='Find Files', command=self.cal_button1)

self.reactor1_wt_button.grid(row=0, column=2)

self.reactor2_wt_button = ttk.Button(self.calibration_label, text='Find Files', command=self.cal_button2)
self.reactor2_wt_button.grid(row=1, column=2)

self.reactor3_wt_button = ttk.Button(self.calibration_label, text='Find Files',
command=self.cal_button3)

self.reactor3_wt_button.grid(row=2, column=2)

self.reactor4_wt_button = ttk.Button(self.calibration_label, text='Find Files',
command=self.cal_button4)

self.reactor4_wt_button.grid(row=3, column=2)

def show_page_event(self, event):

    self.port_combobox['values'] = [port.device for port in port_list.comports()]

    global _serial_connection

    if _serial_connection.isOpen():

        _serial_connection.close()

    global real_time_graph

    real_time_graph.event_source.stop()

    global motor_calibration_graph

    motor_calibration_graph.event_source.stop()

    global weight_calibration_graph
weight_calibration_graph.event_source.stop()

def upload_code(self):
    
names = {'Main Program': 'main', 'Motor Calibration': 'motor', 'Weight Calibration': 'weight'}

    global update_rtc

    if self.main_prog_update_rtc.get():
        update_rtc = True
    else:
        update_rtc = False

    if self.main_prog_update_start_time.get():
        writeMotorParams(parameter_list, pid_dict)
    else:
        with open(_cli_args.directory +
'src/arduino/libraries/ReactorControl/ReactorConstants.h', 'r') as reactor_constants:
            all_lines = reactor_constants.readlines()

            start_time = all_lines[2].split(' ')[2]

        writeMotorParams(parameter_list, pid_dict, start_time)
_cli_args.code_base = names[self.prog_selection.get()]

self.upload_error = uploadCode(_cli_args.code_base, _cli_args.port, _cli_args.directory)

if self.upload_error.returncode:
    msgbx.showerror(title='Upload Error', message=self.upload_error.stderr)
else:
    msgbx.showinfo(title='Upload Complete', message='Upload Completed Successfully!)

def show_graph_page(self):
    if self.prog_selection.get() == 'Main Program':
        self.controller.show_page(DataGraph)
    elif self.prog_selection.get() == 'Motor Calibration':
        self.controller.show_page(MotorCalibration)
    elif self.prog_selection.get() == 'Weight Calibration':
        self.controller.show_page(WeightCalibration)

def update_port(self, event=None):
global _cli_args

self.port_combobox.config(values=[port.device for port in port_list.comports()])

_cli_args.port = self.port_selection.get()

def cal_button1(self):

    self.reactor_cal_files[0].set(tk.filedialog.askopenfilenames(title='Select calibration files',
        filetypes = (('Calibration Files', '*.wt *.cal'), ('All Files', '*.*')),
        initialdir='~/Dropbox/Lab/Bioreactor/Arduino_Code/calibration_file'))


def cal_button2(self):

    self.reactor_cal_files[1].set(tk.filedialog.askopenfilenames(title='Select calibration files',
        filetypes = (('Calibration Files', '*.wt *.cal'), ('All Files', '*.*')),
        initialdir='~/Dropbox/Lab/Bioreactor/Arduino_Code/calibration_file'))


def cal_button3(self):
self.reactor_cal_files[2].set(tk.filedialog.askopenfilenames(title='Select calibration files',
filetypes=((('Calibration Files', '*.wt *.cal'), ('All Files', '*.*')),
initialdir='~/Dropbox/Lab/Bioreactor/Arduino_Code/calibration_file'))

def cal_button4(self):
    self.reactor_cal_files[3].set(tk.filedialog.askopenfilenames(title='Select calibration files',
filetypes=((('Calibration Files', '*.wt *.cal'), ('All Files', '*.*')),
initialdir='~/Dropbox/Lab/Bioreactor/Arduino_Code/calibration_file'))

def calc_calibration(self):
    pass

class CommandCenter(tk.Tk):
    def __init__(self, startPage=Uploader, *args, **kwargs):
        tk.Tk.__init__(self, *args, **kwargs)
        tk.Tk.wm_title(self, "Bioreactor Command Center")
self.tk.call('wm', 'iconphoto', self._w,
            tk.PhotoImage(file=('/home/lukas/Dropbox/Lab/Bioreactor/src/python/flask_icon.png')))
def writeMotorParams(parameter_lists, pid_vals, start_time=None, start_high=False, directory=None, update_rtc=None):

    '''
    This function takes a list of four dictionaries containing the parameters needed to run the IVD bioreactor.

    :param parameter_list: listlike containing four dictionaries with 11 parameters.

    parameter_list = [reactor0_profile, reactor1_profile, reactor2_profile, reactor3_profile]

    reactor0_profile = {'high_mode_on_time': 7200, #the number of seconds per 24hr period that the reactor will spend in high mode
'high_mode_frequency' : 1000, #the number of milliseconds between min and max

'high_mode_max_load' : 7396535, #load cell reading for max load

'high_mode_min_load' : 5795344, #load cell reading for min load

'low_mode_frequency' : 1000, #the number of milliseconds between min and max

'low_mode_max_load' : 6595940, #load cell reading for max load

'low_mode_min_load' : 5795344, #load cell reading for min load

'error_bounds' : 60000, #the +/- that is acceptable between load cell reading and setpoint

'load_cell_offset' : 100000} #the offset needed so the load cell reading doesn't underflow while the 'potato masher' is hanging

:param pid_vals: The PID values for the reactor system as a dict.

:param start_time: unix timestamp of when the experiment began.

default is to get the current time.
:param start_high: boolean determining whether the arduino program starts in
mode, otherwise it will start in low mode.

:param directory: a string containing the address of the reactor constants file.

```
# some checks to make sure all of the parameters have a value or the arduino program
won't compile

if len(parameter_lists) < 4:
    raise ValueError('parameter_list is too short, need all 4 parameter lists with 9 values
each (placeholder padding is fine')

for dict_of_params in parameter_lists:
    if len(dict_of_params) < 9:
        raise ValueError('parameter_list is too short, need all 4 parameter lists with 9
values each (placeholder padding is fine')

    # To the person using this in the future change the default directory to the one you
are using

    if directory is not None:
        directory = directory

else:

directory =
'/home/lukas/Dropbox/arduino/libraries/ReactorControl/ReactorConstants.h'

# a quick change from python boolean to C boolean

if start_high:
    start_high = 'true'
else:
    start_high = 'false'

with open(directory, 'w') as reactor_profile:
    print('#ifndef ReactorConstants_h', file=reactor_profile)
    print('#define ReactorConstants_h', file=reactor_profile)
    if start_time is not None:
        print('#define EXPERIMENT_START_TIME {}
'.format(int(start_time)), file=reactor_profile)
    else:
        print('#define EXPERIMENT_START_TIME {}
'.format(int(time.time())), file=reactor_profile)
    if update_rtc is not None:
print('#define UPDATE_RTC true', file=reactor_profile)

else:

print('#define UPDATE_RTC false', file=reactor_profile)

print('#define START_IN_HIGH_MODE {'.format(start_high), file=reactor_profile)

for index, dict_of_params in enumerate(parameter_lists):

print('#define R{}_HIGH_MODE_ON_TIME {high_mode_on_time}'.format(index, **dict_of_params), file=reactor_profile)

print('#define R{}_HIGH_MODE_FREQUENCY {high_mode_frequency}'.format(index, **dict_of_params), file=reactor_profile)
print('#define R{}_HIGH_MODE_MAX_LOAD
{high_mode_max_load}'.format(index, **dict_of_params), file=reactor_profile)

print('#define R{}_HIGH_MODE_MIN_LOAD
{high_mode_min_load}'.format(index, **dict_of_params), file=reactor_profile)

print('#define R{}_LOW_MODE_FREQUENCY
{low_mode_frequency}'.format(index, **dict_of_params), file=reactor_profile)

print('#define R{}_LOW_MODE_MAX_LOAD
{low_mode_max_load}'.format(index, **dict_of_params), file=reactor_profile)

print('#define R{}_LOW_MODE_MIN_LOAD
{low_mode_min_load}'.format(index, **dict_of_params), file=reactor_profile)

print('#define R{}_ERROR_BOUNDS {error_bounds}'.format(index, **dict_of_params), file=reactor_profile)

print('#define R{}_LOAD_CELL_OFFSET {load_cell_offset}
'.format(index, **dict_of_params), file=reactor_profile)

print('#endif', file=reactor_profile)


def uploadCode(code_file='main', port=None, directory=None, verify=False):
    
    

This function will handle code uploads to the arduino:

:param code_file: 'main' | 'weight' | 'motor'

:param directory: 'directory address of arduino program folder'

:param port: 'arduino port location'

if not((code_file == 'main') or (code_file == 'weight') or (code_file == 'motor')):
    raise(ValueError('code_file must be "main", "weight", or "motor"'))
else:
    code_file = code_file

if not verify:
    if port is not None:
        port = port
    if port not in [port.device for port in port_list.comports()]:
        print([port.device for port in port_list.comports()])
        print('The default port is not available. Please try another from those listed above."
        port = input('new port>>')
    else:
if '/dev/ttyACM0' in [port.device for port in port_list.comports()]:

    port = '/dev/ttyACM0'

else:

    print([port.device for port in port_list.comports()])

    print('The default port is not available. Please try another from those listed above. ')

    port = input('new port>>')

if directory is not None:

    directory = directory

else:

    directory = '/home/lukas/Dropbox/Lab/Bioreactor/

if code_file == 'motor':

    program_location = 'src/arduino/motorCalibration/motorCalibration.ino'

elif code_file == 'weight':

    program_location = 'src/arduino/weightCalibration/weightCalibration.ino'

else:

    program_location = 'src/arduino/BioreactorController/BioreactorController.ino'

if verify:
result = sub.run(['arduino',
                  '--verify', directory + program_location,
                  '--board','arduino:avr:mega:cpu=atmega2560'],
                  # stdout=sub.PIPE,
                  # stderr=sub.PIPE,
                  encoding='utf-8')

else:

    result = sub.run(['arduino',
                      '--upload', directory + program_location,
                      '--port', port,
                      '--board','arduino:avr:mega:cpu=atmega2560'],
                      stdout=sub.PIPE,
                      stderr=sub.PIPE,
                      encoding='utf-8')

    return result

def weight_calibration(calibration_file):

calibration_df = pd.read_csv(calibration_file, header=None, names=['weight', 'reading'])

weight_regression = LinearRegression().fit(calibration_df['reading'].values.reshape(-1,1),
                                           calibration_df['weight'].values.reshape(-1,1))

return (weight_regression.coef_[0][0], weight_regression.intercept_[0],
        weight_regression.score(calibration_df['reading'], calibration_df['weight']))

reactor0_profile = {'high_mode_on_time' : 28800,
                     'high_mode_frequency' : 2000,
                     'high_mode_max_load' : 8200000,
                     'high_mode_min_load' : 6821739,
                     'low_mode_frequency' : 2000,
                     'low_mode_max_load' : 5800210,
                     'low_mode_min_load' : 5800210,
                     'error_bounds' : 60000,
                     'load_cell_offset' : 100000}
reactor0_profile = {'high_mode_on_time' : 28800,
                   'high_mode_frequency' : 2000,
                   'high_mode_max_load' : 4859545,
                   'high_mode_min_load' : 2466336,
                   'low_mode_frequency' : 2000,
                   'low_mode_max_load' : 1269732,
                   'low_mode_min_load' : 1269732,
                   'error_bounds' : 60000,
                   'load_cell_offset' : 100000}

reactor1_profile = {'high_mode_on_time' : 22800,
                   'high_mode_frequency' : 2000,
                   'high_mode_max_load' : 2968204,
                   'high_mode_min_load' : 1033168,
                   'low_mode_frequency' : 2000,
                   'low_mode_max_load' : 2968204,
                   'low_mode_min_load' : 1033168,
                   'error_bounds' : 60000,
reactor2_profile = {
    'high_mode_on_time': 28800,
    'high_mode_frequency': 2000,
    'high_mode_max_load': 8200000,
    'high_mode_min_load': 6944856,
    'low_mode_frequency': 1000,
    'low_mode_max_load': 5675923,
    'low_mode_min_load': 5675923,
    'error_bounds': 60000,
    'load_cell_offset': 100000
}

# reactor2_profile = {
#    'high_mode_on_time': 28800,
#    'high_mode_frequency': 2000,
#    'high_mode_max_load': 3640226,
#    'high_mode_min_load': 1921935,
#    'low_mode_frequency': 1000,
#    'low_mode_max_load': 1062789,
#    'low_mode_min_load': 1062789,
reactor3_profile = {
    'high_mode_on_time' : 22800,
    'high_mode_frequency' : 2000,
    'high_mode_max_load' : 7396535,
    'high_mode_min_load' : 5795344,
    'low_mode_frequency' : 1000,
    'low_mode_max_load' : 6595940,
    'low_mode_min_load' : 5795344,
    'error_bounds' : 60000,
    'load_cell_offset' : 100000
}

parameter_list = [reactor0_profile, reactor1_profile, reactor2_profile, reactor3_profile]

pid_dict = {
    'p_val_fwd' : 50000,
    'i_val_fwd' : 5000000,
    'p_val_rev' : 500000,
    'i_val_rev' : 50000000
}
def main():

    import argparse

    parser = argparse.ArgumentParser()

    parser.add_argument('-u', '--upload',
                        nargs='?',
                        const='upload',
                        choices=['u', 'upload', 'r', 'reupload'],
                        help='uploads the code updating the experiment start time and other parameters.\n
Use option "reupload" to skip updating the start time and parameters.')

    parser.add_argument('-v', '--verify',
                        action='store_true',
                        help='verifies(compiles) the selected codebase(default=main), does not upload')

    parser.add_argument('-c', '--code-base',
                        choices=['main', 'weight', 'motor'],
                        default='main',
                        help='specifies version of the bioreactor program to upload, [main, weight, motor]')
parser.add_argument('-p', '--port',
    default='/dev/ttyACM0',
    help='selects which port to upload the code to')

parser.add_argument('-d', '--directory',
    default= '/home/lukas/Dropbox/Lab/Bioreactor/',
    help='the root directory where the code base is stored')

parser.add_argument('-o','--open-cc',
    action='store_true',
    help='opens the command center which is a gui for the bioreactor, this will launch
    if no arguments are passed to the bioreactor program')

parser.add_argument('-g','--open-graph',
    action='store_true',
    help='opens a matplotlib graph with the setpoint and current loading of the
    reactors')

    global _cli_args

    global _serial_connection
cli_args = parser.parse_args()

serial_connection = serial.Serial(timeout=0.1)

serial_connection.port = cli_args.port

serial_connection.baudrate = 115200

print('Current Home Directory is "{}"
'.format(cli_args.directory))

print('Current Code Base is : {}
'.format(cli_args.code_base))

with open(cli_args.directory+'data/config/bioreactor_profiles.yaml', 'r') as bioreactor_profile_parameters:
    parameter_list = yaml.load(bioreactor_profile_parameters)

if len(sys.argv) == 1 or cli_args.open_cc:
    print('Opening Comamnd Center GUI')
    cc = CommandCenter()
    cc.mainloop()

if _cli_args.upload == 'upload':
print('Uploading code base: {}\nTo port: {}\n'.format(_cli_args.code_base, _cli_args.port))

writeMotorParams(parameter_list, pid_dict)

uploadCode(_cli_args.code_base, _cli_args.port, _cli_args.directory)

print('Upload Successful!')

elif _cli_args.upload == 'reupload' or _cli_args.upload == 'r':

    print('Reuploading code base: {}\nTo port: {}\n'.format(_cli_args.code_base, _cli_args.port))

    uploadCode(_cli_args.code_base, _cli_args.port, _cli_args.directory)

    print('Reupload Successful!')

if _cli_args.verify:

    print('Verifying code base: {}\n'.format(_cli_args.code_base))

    uploadCode(_cli_args.code_base, _cli_args.port, _cli_args.directory, verify=True)

if _cli_args.open_graph == True:

    print('Opening Real Time Graphing Utility')

    cc = CommandCenter(DataGraph)

    cc.mainloop()

_serial_connection.close()
if __name__ == '__main__':

    main()

ADC Library

Header files

NAU7802.h

/*

* NAU7802 Library

*.h file

*

* This library is used to simplify the control scheme of the NAU7802 bridge sensor chip for use with a bioreactor controller.

* this chip has many registers and options not all of which are explained well in the data-sheet. As far as I am aware all register settings are expressed here,

* (The direct OTP read setting is hidden from the user though) if any are missing feel free to log it.
**IMPORTANT**

To start the chip up a specific sequence of commands needs to be used to ensure that the IC starts up properly and is ready to use.

After the IC receives power the recommended start up sequence is: ->Can use bootCycle function instead

* NAU7802::registerReset(true);

* NAU7802::registerReset(false); // This register value needs to be turned off for normal chip operation

* NAU7802::powerDigital(); // the default value is true so it should turn on

* delay(1); // the IC needs at least 200usec to boot up its internal circuitry

* NAU7802::powerAnalog(); // Powers up the ADC

* {Set Gain, Offsets, and all other desired settings}

* {Calibrate}

* NAU7802::cycleStart(); // ***This triggers the first conversion and needs to be activated NOTE: It can take a few milliseconds for the first conversion to initialize

* NOTE: The IC has 50kOhm I2C pull-ups enabled internally by default, and can use 1.6kOhm ones, but this must be set in the registers.
// **NOTE: the function Wire.begin() will need to be called somewhere in the
// main body of your program, BEFORE you call any of the library functions

#ifndef NAU7802_H
#define NAU7802_H

#include "Arduino.h"

#include <Wire.h>
//#define LEAN true //This library is pretty large and if space is tight it cuts out a lot of
functions and data structures to the bare minimum to save space (down to 3 functions and
1 data structure)

#ifndef LEAN

typedef union Offsets { //IC reading offsets, can run calibration or enter manually
    uint32_t Offset;
    uint8_t OffBytes[3];
} Offsets;

typedef union GOffsets { //Gain offset values, can run a calibration or set manually
    uint32_t GOffset;
    uint8_t GOffBytes[4];
} GOffsets;

typedef union OTPread { //Internal temperature sensor values
    uint32_t OTPvalue;
} OTPread;
typedef enum gainSelect { //used to select the desired Gain into the ADC
    x1 = 0b000,
    x2 = 0b001,
    x4 = 0b010,
    x8 = 0b011,
    x16 = 0b100,
    x32 = 0b101,
    x64 = 0b110,
    x128 = 0b111
} gainSelect;

typedef enum LDOSelect { //used to select the internal IC LDO output
    V45 = 0,   //4.5V
    V42,       //4.2V
V39,

V36,

V33,

V30,

V27,

V24  //2.4V

} LDOSelect;

typedef enum sampleRate { //Used in setting the sample rate (samples per second)
    SPS10 = 0b000,
    SPS20,
    SPS40,
    SPS80,
    SPS320 = 0b111
} sampleRate;

typedef enum calType { //used to set which type of calibration will be run
INTERN = 0b00,
OFFSET = 0b10,
GAIN = 0b11
}
calType;

typedef enum ComMode { //used to set the common mode
    DISABLED = 0b00,
    REFN = 0b10,
    REFP = 0b11
} ComMode;

typedef enum I2C_level { //used to set the integrates I2C pullups
    NOPULLUP = 0b00,
    WEAK = 0b10,
    STRONG = 0b11
} I2C_level;
typedef enum registerAddress { //Register address map, used in selecting which address is being worked with. NOTE: ADC_REG and OTP_READ share the 0x15 register and to read the state the proper mode must be selected, any writes always go to the ADC_REG

    PU_CTRL     = 0x00,

    CTRL1       = 0x01,

    CTRL2       = 0x02,

    CH1_OFFSET  = 0x03,

    CH1_GOFFSET = 0x06,

    CH2_OFFSET  = 0x0A,

    CH2_GOFFSET = 0x0D,

    I2C_CTRL    = 0x11,

    ADC_RESULT  = 0x12,

    ADC_REG     = 0x15,

    OTP_READ    = 0x15,

    PGA_REG     = 0x1B,

    PWR_CTRL    = 0x1C

} registerAddress;
typedef union loadReading { //the load reading taken from the IC
    int32_t Load;
    uint8_t bytes[3];
} loadReading;

class NAU7802 {

    public:

    NAU7802();

    #ifndef LEAN

    static void selectAVDDsource(bool internal_source = false); // select the AVdd source
    (external = 0 (default) internal LDO =1)
static void selectClockSource(bool internal_source = true); //select the IC clock source
(internal = 0 (default) external = 1)

static void cycleStart(bool start = true); //starts the next conversion at the rising edge of
this trigger. Needed to start the first conversion.

static bool powerAnalog(bool powerUp = true); //power up the analog circuitry

static bool powerDigital(bool powerUp = true); //power up all digital circuitry in the IC

static bool registerReset(bool reset = true); //resets all registers to their default value.
NOTE: This bit must be manually cleared otherwise the chip will reset itself indefinitely.

static bool checkPowerUp(); //checks the power-up status of the IC to see if it is ready
to use

static bool checkDataReady(); //checks the register to see if new data is ready

static void setRDYpinPolarity(bool activeHigh = true); //sets the data conversion pin
polarity (default HIGH = conversion ready)

static uint8_t setGain(gainSelect gain); //Set the ADC gain from 1-128

static void setVLDO(LDOSelect VLDO); //Set the AVDD/LDO voltage from 4.5V-2.4V

static void setChannel(uint8_t channel); //select channel 1 (0) or channel 2 (1)

static void setConversionRate(sampleRate conversionRate); //Set the Samples per
second 10-320
static void setCalibrationMode(calType calibrationSelect);//selects what type of
 calibration will be run during the calibrate function

static bool calibrate();//runs a calibration, what is being calibrated is set by
 setCalibrationMode()

static void setSDAconvRDY(bool enabled = true); //special non-standard I2C protocol
which allows the IC to use the SDA line to be pulled low to signal conversion
completion, if the data line is not in use

static void setFastRead(bool enabled = true); //special I2C protocol which allows the IC
to convert out all 24 to 32 bits of the desired data in a "burst mode" with no ACKs
between bytes

static void I2CstrPU(bool enabled = true); //Use the strong pull-ups on the I2C line
(they are 1.6k)

static void I2CwkPU(bool enabled = true); //Use the weak I2C pull-ups (50K). These
pull-ups are enabled by default and must be disabled to be turned off

static void shortInput(bool enabled = true); //internally shorts the input lines together
and measures the offset (where this value is stored or used is unclear in the datasheet)

static void setBurnoutCurrent(bool enabled = true); //enables the burnoutcurrent on the
PGA positive input

static void setTempSensor(bool enabled = true); //sets the PGA to read from the temp
sensor vs input
static void setBandgapChopper(bool enabled = true); //enables or disables the band-gap chopper

static void setCLKfreq(uint8_t freq); // if set to 0b11 will be turned off otherwise on (not explained in datasheet)

static void setCommonMode(ComMode mode);//For single ended operation, sets which pin will be the common mode.

static void setChopClockDelay(uint8_t select);

static uint32_t readOTP(); //read the IC temperature sensor

static void setLDOmode(bool highESR);

static void setPGAoutputBuffer(bool enabled = true);

static void setPGAbypass(bool enabled = true);

static void setPGAinputInversion(bool enabled = true);

static void setPGAchopper(bool enabled = false);

static void setPGAbypassCAP(bool enabled = true);

static void setMasterBiasCurrent(uint8_t current);

static void setADCcurrent(uint8_t current);

static void setPGAcurrent(uint8_t current);

static void setLoadCellOffset(Offsets offset);
static void bootCycle(void);//runs through the boot sequence needed to start the
NAU7802

static Offsets readOffset(uint8_t channel = 1);

static GOffsets readGOffset(uint8_t channel = 1);

static uint8_t readRegister(registerAddress address); // read the current value of any register

static void setRegister(registerAddress address, uint8_t regValue); //set the value of any register directly (use the datasheet)

#endif

static int32_t readLoad(); //reads the load and returns the value as a 32-bit integer (the result only has 24bits)

static uint8_t readRegister(uint8_t address); // read the current value of any register

static void setRegister(uint8_t address, uint8_t regValue); //set the value of any register directly (use the datasheet)

private:
static const uint8_t NAU7802_ADDRESS = 0x2A;

};

#endif

Source
NAU7802.cpp

/*

* NAU7802 Library

* .cpp file

*

* This library is used to simplify the control scheme of the NAU7802 bridge sensor chip for use with a bioreactor controller.

* This library holds the current register state in one structure meaning it is relatively large, this is done so that the current register state does not have to be read before changing it increasing communication speed.

* As such if space is a concern the library can be cut down by defining the macro LEAN to true (honestly any value will do).

*

* Lukas Jaworski

*
void NAU7802::selectAVDDsource(bool internal_source){
    // Allows the selection of the analog voltage source

    uint8_t currentRegister = NAU7802::readRegister(PU_CTRL);

    currentRegister = ((currentRegister & 0b01111111) | (internal_source << 7));

    NAU7802::setRegister(PU_CTRL, currentRegister);

    return;
}
void NAU7802::selectClockSource(bool internal_source) { //Decides whether the internal or an external clock is used

    uint8_t currentRegister = NAU7802::readRegister(PU_CTRL);
    currentRegister = ((currentRegister & 0b10111111) | (internal_source << 6));
    NAU7802::setRegister(PU_CTRL, currentRegister);
    return;
}

void NAU7802::cycleStart(bool start) {

    uint8_t currentRegister = NAU7802::readRegister(PU_CTRL);
    currentRegister = ((currentRegister & 0b11101111) | (start << 4));
    NAU7802::setRegister(PU_CTRL, currentRegister);
    return;
}

bool NAU7802::powerAnalog(bool powerUp) {

    uint8_t currentRegister = NAU7802::readRegister(PU_CTRL);

currentRegister = ((currentRegister & 0b11111011) | (powerUp << 2));

NAU7802::setRegister(PU_CTRL, currentRegister);

return powerUp;
}

bool NAU7802::powerDigital(bool powerUp){

  uint8_t currentRegister = NAU7802::readRegister(PU_CTRL);
  currentRegister = ((currentRegister & 0b11111101) | (powerUp << 1));
  NAU7802::setRegister(PU_CTRL, currentRegister);
  return powerUp;
}

bool NAU7802::registerReset(bool reset){

  uint8_t currentRegister = NAU7802::readRegister(PU_CTRL);
  currentRegister = ((currentRegister & 0b11111110) | reset);
  NAU7802::setRegister(PU_CTRL, currentRegister);
  return reset;
bool NAU7802::checkPowerUp()
{
    char currentRegister = NAU7802::readRegister(PU_CTRL);
    if (currentRegister == -1)
    {
        return false;
    }
    if (((currentRegister >> 3) & 1))
    {
        return true;
    }
    else
    {
        return false;
    }
}

bool NAU7802::checkDataReady()
{
    char currentRegister = NAU7802::readRegister(PU_CTRL);
if (currentRegister == -1) {
    return false;
}

else if (((currentRegister >> 5) & 1)) {
    return true;
}

else {
    return false;
}

uint8_t NAU7802::setGain(gainSelect gain) {
    uint8_t currentRegister = NAU7802::readRegister(CTRL1);
    currentRegister = ((currentRegister & 0b11111000) | gain);
    NAU7802::setRegister(CTRL1, currentRegister);
    return uint8_t(gain);
}
void NAU7802::setVLDO(LDOSelect VLDO) {
    uint8_t currentRegister = NAU7802::readRegister(CTRL1);
    currentRegister = ((currentRegister & 0b11000111) | (VLDO << 3));
    NAU7802::setRegister(CTRL1, currentRegister);
    return;
}

void NAU7802::setChannel(uint8_t channel) {
    uint8_t currentRegister = NAU7802::readRegister(CTRL2);
    currentRegister = ((currentRegister & 0b01111111) | (channel << 7));
    NAU7802::setRegister(CTRL2, currentRegister);
    return;
}

void NAU7802::setConversionRate(sampleRate conversionRate) {
    uint8_t currentRegister = NAU7802::readRegister(CTRL2);

currentRegister = ((currentRegister & 0b10001111) | (conversionRate << 4));

NAU7802::setRegister(CTRL2, currentRegister);

return;

void NAU7802::setCalibrationMode(calType calibrationSelect) {

  uint8_t currentRegister = NAU7802::readRegister(CTRL2);
  
  currentRegister = ((currentRegister & 0b11111100) | (calibrationSelect));

  NAU7802::setRegister(CTRL2, currentRegister);

  return;

}

bool NAU7802::calibrate() { //starts a calibration, waits until it is complete and returns true if there were no errors, throws false on a calibration error

  uint8_t currentRegister = NAU7802::readRegister(CTRL2);

  currentRegister = ((currentRegister & 0b11111100) | (calibrationSelect));

  NAU7802::setRegister(CTRL2, currentRegister);

  do {
    
    // calibration logic here
    
    // check for completion
    
    // if complete, return true
    
    // if error, return false
    
  } while (/* condition for ongoing calibration */);

  return true;  // calibration successful

}
currentRegister = NAU7802::readRegister(CTRL2);

} while( 1 & (currentRegister>>2));

if ( 1 & (currentRegister>>3)) {return false;}
else {return true;}

}

void NAU7802::setSDAconvRDY(bool enabled){ //allow the SDA to become a conversion ready interrupt when not in use. (not standard I2C)

uint8_t currentRegister = NAU7802::readRegister(I2C_CTRL);

currentRegister = ((currentRegister & 0b01111111) | (enabled << 7));

NAU7802::setRegister(I2C_CTRL, currentRegister);

return;

}

void NAU7802::setFastRead(bool enabled){ //allow fast read mode of ADC data, ADC_REG bit 7 must also be 1  (not standard I2C)

uint8_t currentRegister = NAU7802::readRegister(I2C_CTRL);

currentRegister = ((currentRegister & 0b10111111) | (enabled << 6));


NAU7802::setRegister(I2C_CTRL, currentRegister);

currentRegister = NAU7802::readRegister(ADC_REG);

currentRegister = ((currentRegister & 0b01111111) | (enabled << 7));

NAU7802::setRegister(ADC_REG, currentRegister);

return;

} // void NAU7802:: I2CpullUps(I2C_level I2Clevel){

// uint8_t currentRegister = NAU7802::readRegister(I2C_CTRL);

// switch (I2Clevel){

// case NOPULLUP:

// break;

// case WEAK:

// currentRegister = ((currentRegister & 0b11101111) | ((~1) << 4));

// break;

// case STRONG:

// currentRegister = ((currentRegister & 0b11001111) | (1 << 5));
// break;

// }

// NAU7802::setRegister(I2C_CTRL, currentRegister);

// return;

// }

void NAU7802::I2CstrPU(bool enabled){ //use strong pullup resistors on the i2c (1.6k)

uint8_t currentRegister = NAU7802::readRegister(I2C_CTRL);

currentRegister = ((currentRegister & 0b11011111) | (enabled << 5));

NAU7802::setRegister(I2C_CTRL, currentRegister);

return;
}

void NAU7802::I2CwkPU(bool enabled){ //use weak pullup resistors on the i2c (50k)
(These are enabled by default)

uint8_t currentRegister = NAU7802::readRegister(I2C_CTRL);

currentRegister = ((currentRegister & 0b11101111) | ((~enabled & 1) << 4));

NAU7802::setRegister(I2C_CTRL, currentRegister);
void NAU7802::shortInput(bool enabled) { //short the input and measure the offset
    uint8_t currentRegister = NAU7802::readRegister(I2C_CTRL);
    currentRegister = ((currentRegister & 0b11110111) | (enabled << 3));
    NAU7802::setRegister(I2C_CTRL, currentRegister);
    return;
}

void NAU7802::setBurnoutCurrent(bool enabled) { //I really don't know what the burnout
current is or does, but the setting is in the data sheet so you can set it here.
    uint8_t currentRegister = NAU7802::readRegister(I2C_CTRL);
    currentRegister = ((currentRegister & 0b11111011) | (enabled << 2));
    NAU7802::setRegister(I2C_CTRL, currentRegister);
    return;
}
void NAU7802::setTempSensor(bool enabled) { // sets the temp sensor to go to the PGA
    uint8_t currentRegister = NAU7802::readRegister(I2C_CTRL);
    currentRegister = ((currentRegister & 0b11111101) | (enabled << 1));
    NAU7802::setRegister(I2C_CTRL, currentRegister);
    return;
}

void NAU7802::setBandgapChopper(bool enabled) { // enables the bandgap chopper (this is enabled by default)
    uint8_t currentRegister = NAU7802::readRegister(I2C_CTRL);
    currentRegister = ((currentRegister & 0b11111110) | (~enabled & 1));
    NAU7802::setRegister(I2C_CTRL, currentRegister);
    return;
}

void NAU7802::setCLKfreq(uint8_t freq) {
    uint8_t currentRegister = NAU7802::readRegister(ADC_REG);
    currentRegister = ((currentRegister & 0b11001111) | (freq<<4));
NAU7802::setRegister(ADC_REG, currentRegister);

return;
}

void NAU7802::setCommonMode(ComMode mode) {
    uint8_t currentRegister = NAU7802::readRegister(ADC_REG);

    currentRegister = ((currentRegister & 0b11110011) | (mode << 2));

    NAU7802::setRegister(ADC_REG, currentRegister);

    return;
}

void NAU7802::setChopClockDelay(uint8_t select) { // the datasheet has nothing listed about these modes

    uint8_t currentRegister = NAU7802::readRegister(ADC_REG);

    currentRegister = ((currentRegister & 0b11111100) | (select));

    NAU7802::setRegister(ADC_REG, currentRegister);

    return;
}
uint32_t NAU7802::readOTP(){ //ADC_REG and the MSB of the OTP_READ share a
register address and a setting needs changed to read the OTP, this function sets that
setting and then reverts back to reading the ADC_REG value

    OTPread currentTemp;

    uint8_t currentRegister = NAU7802::readRegister(PGA_REG);

    currentRegister = (currentRegister | (1<<7));

    NAU7802::setRegister(PGA_REG, currentRegister);

    Wire.beginTransmission(NAU7802_ADDRESS);

    Wire.write(uint8_t (OTP_READ));

    Wire.endTransmission(false);

    Wire.requestFrom(NAU7802_ADDRESS, uint8_t(4));

    for (int8_t i = 3; i >= 0; i--){

        currentTemp.OTPbytes[i] = Wire.read();

    }

    Wire.endTransmission();

    currentRegister = NAU7802::readRegister(PGA_REG);

    currentRegister = (currentRegister & ~(1<<7));
NAU7802::setRegister(PGA_REG, currentRegister);

return currentTemp.OTPvalue;

}

void NAU7802::setLDOmode(bool highESR) {

    uint8_t currentRegister = NAU7802::readRegister(PGA_REG);

    currentRegister = ((currentRegister & 0b10111111) | (highESR << 6));

    NAU7802::setRegister(PGA_REG, currentRegister);

    return;

}

void NAU7802::setPGAoutputBuffer(bool enabled) {

    uint8_t currentRegister = NAU7802::readRegister(PGA_REG);

    currentRegister = ((currentRegister & 0b11011111) | (enabled << 5));

    NAU7802::setRegister(PGA_REG, currentRegister);

    return;

}
void NAU7802::setPGAbypass(bool enabled) {
    uint8_t currentRegister = NAU7802::readRegister(PGA_REG);

    currentRegister = ((currentRegister & 0b11101111) | (enabled << 4));

    NAU7802::setRegister(PGA_REG, currentRegister);

    return;
}

void NAU7802::setPGAinputInversion(bool enabled) {
    uint8_t currentRegister = NAU7802::readRegister(PGA_REG);

    currentRegister = ((currentRegister & 0b11110111) | (enabled << 3));

    NAU7802::setRegister(PGA_REG, currentRegister);

    return;
}

void NAU7802::setPGAchopper(bool enabled) {
    uint8_t currentRegister = NAU7802::readRegister(PGA_REG);

    currentRegister = ((currentRegister & 0b11111111) | (enabled << 7));

    NAU7802::setRegister(PGA_REG, currentRegister);

    return;
}
currentRegister = ((currentRegister & 0b111111110) | (~enabled & 1));

NAU7802::setRegister(PGA_REG, currentRegister);

return;

}

void NAU7802::setPGAbypassCAP(bool enabled){

  uint8_t currentRegister = NAU7802::readRegister(PWR_CTRL);

  currentRegister = ((currentRegister & 0b01111111) | (enabled << 7));

  NAU7802::setRegister(PWR_CTRL, currentRegister);

  return;

}

void NAU7802::setMasterBiasCurrent(uint8_t current){

  uint8_t currentRegister = NAU7802::readRegister(PWR_CTRL);

  currentRegister = ((currentRegister & 0b10001111) | (current << 4));

  NAU7802::setRegister(PWR_CTRL, currentRegister);

  return;
void NAU7802::setADCcurrent(uint8_t current) {
    uint8_t currentRegister = NAU7802::readRegister(PWR_CTRL);
    currentRegister = ((currentRegister & 0b11110011) | (current << 2));
    NAU7802::setRegister(PWR_CTRL, currentRegister);
    return;
}

void NAU7802::setPGAcurrent(uint8_t current) {
    uint8_t currentRegister = NAU7802::readRegister(PWR_CTRL);
    currentRegister = ((currentRegister & 0b01111100) | (current));
    NAU7802::setRegister(PWR_CTRL, currentRegister);
    return;
}

void NAU7802::setLoadCellOffset(Offsets offset) {
//TODO make it convert negative numbers properly

NAU7802::setRegister(0x05, offset.OffBytes[0]);

NAU7802::setRegister(0x04, offset.OffBytes[1]);

NAU7802::setRegister(0x03, offset.OffBytes[2]);

}

void NAU7802::bootCycle(void){

NAU7802::registerReset(true);

delay(1);

NAU7802::registerReset(false);

NAU7802::powerDigital();

delay(1);

NAU7802::powerAnalog();

}

Offsets NAU7802::readOffset(uint8_t channel){

Offsets currentOffset;

Wire.beginTransmission(NAU7802_ADDRESS);

if (channel == 1) {
    Wire.write(uint8_t (CH1_OFFSET));
} else if (channel == 2) {
    Wire.write(uint8_t (CH2_OFFSET));
} else {
    Wire.endTransmission();
    currentOffset.Offset = -1;
    return currentOffset;
}

Wire.endTransmission(false);

Wire.requestFrom(NAU7802_ADDRESS, uint8_t(3));

for (int8_t i = 2; i >= 0; i--){
    currentOffset.OffBytes[i] = Wire.read();
}

Wire.endTransmission();
GOffsets NAU7802::readGOffset(uint8_t channel) {

    GOffsets currentGOffset;

    Wire.beginTransmission(NAU7802_ADDRESS);

    if (channel == 1) {
        Wire.write(uint8_t (CH1_GOFFSET));
    } else if (channel == 2) {
        Wire.write(uint8_t (CH2_GOFFSET));
    } else {
        Wire.endTransmission();
        currentGOffset.GOffset = -1;
        return currentGOffset;
    }

    Wire.endTransmission(false);

    return currentOffset;
}
Wire.requestFrom(NAU7802_ADDRESS, uint8_t(4));

for (int8_t i = 3; i >= 0; i--){
    currentGOffset.GOffBytes[i] = Wire.read();
}

Wire.endTransmission();

return currentGOffset;

}

uint8_t NAU7802::readRegister(registerAddress address){
    return NAU7802::readRegister(uint8_t (address));
}

void NAU7802::setRegister(registerAddress address, uint8_t regValue){
    return NAU7802::setRegister(uint8_t (address), regValue);
}
uint8_t NAU7802::readRegister(uint8_t address){//This function outputs the value stored in one register.
  
  uint8_t registerValue;

  Wire.beginTransmission(NAU7802_ADDRESS);

  Wire.write(address);

  Wire.endTransmission(false);

  Wire.requestFrom(NAU7802_ADDRESS, uint8_t(1));

  registerValue = Wire.read();

  Wire.endTransmission();

  return registerValue;
}

void NAU7802::setRegister(uint8_t address, uint8_t regValue){

  /*
   * This is a generic set register function which will input any 8-bit value into a register
   *
   * The user must take care to only put in valid values as there is no error checking
   */
int32_t NAU7802::readLoad() {

    /*
     * Read the result register and return the value stored there.
     * Negative numbers are correctly read from the result register.
     */

    loadReading currentReading;

    Wire.beginTransmission(NAU7802_ADDRESS);
    Wire.write(uint8_t (ADC_RESULT));
    Wire.endTransmission(false);

    return;
}
Wire.requestFrom(NAU7802_ADDRESS, uint8_t(3));

for (int8_t i = 2; i >= 0; i--){
    currentReading.bytes[i] = Wire.read();
}

Wire.endTransmission();

// Converting negative 24-bit values to negative 32-bit values

if ((currentReading.bytes[2] >> 7) & 1){
    // If the number is negative just pad the MSB with 1s to complete the
    // twos complement negative, easy peasy.
    currentReading.bytes[3] = 0b11111111;
    currentReading.Load = currentReading.Load;
}

return currentReading.Load;
I²C Multiplexer Library

Header files

TCA9544.h

/*

TCA9544 Library

.h file

This library is used to simplify the control scheme of the TCA9544 chip for use with a bioreactor controller.

Lukas Jaworski

University of Miami

2016

*/
//note, the function sd.begin() will need to be called somewhere in the main body of your program

#ifndef TCA9544_H
#define TCA9544_H

#include <Wire.h>

#include "Arduino.h"

class TCA9544 {

public:

TCA9544(uint8_t addressBits = 0);

void changeChannel(uint8_t channelSelect); // The selection is ZERO indexed (0,1,2,3)

uint8_t currentChannel(); // returns the current active channel-actually reads the device control register

uint8_t readInterrupts(); // returns which channels currently have active interrupts

uint8_t readRegister();

}
private:

    uint8_t deviceAddress;

};

#endif

Source
TCA9544.h

/*/ 

TCA9544 Library

.cpp file

This library is used to simplify the controll scheme of the TCA9544 chip for use with a bioreactor controller.

Lukas Jaworski

University of Miami
```cpp
#include <TCA9544.h>

TCA9544::TCA9544(uint8_t addressBits) { // feed in the A0-A2 bits to automatically adjust the TCA's address
    deviceAddress = addressBits + 0x70;
}

void TCA9544::changeChannel(uint8_t channelSelect) {
    Wire.beginTransmission(deviceAddress);
    Wire.write((0b100 | channelSelect));
    Wire.endTransmission();
}

uint8_t TCA9544::readInterrupts() {
    return TCA9544::readRegister() >> 4;
}
```
uint8_t TCA9544::currentChannel() {
    return TCA9544::readRegister() & 0b11;
}

uint8_t TCA9544::readRegister() {
    uint8_t controlRegisterState;

    Wire.requestFrom(deviceAddress, uint8_t(1));

    if (Wire.available()){
        controlRegisterState = Wire.read();
    }

    Wire.endTransmission();

    return controlRegisterState;
}