A NEW AGE: QUANTITATIVE MORPHOMETRY FOR THE 21ST CENTURY – ADVANCES IN THE STUDY OF DORSAL ROOT GANGLION NEURONS

By

MICHAEL BRIAN ANDERSON

Bachelor of Science in Molecular Biology
Northeastern State University
Tahlequah, OK
2012

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 2021

A NEW AGE: QUANTITATIVE MORPHOMETRY FOR THE 21ST CENTURY – ADVANCES IN THE STUDY OF DORSAL ROOT GANGLION NEURONS

Dissertation Approved:

Kenneth E. Miller

Dissertation Adviser

Nedra Wilson

Gerwald Koehler

Aric Warren

ACKNOWLEDGEMENTS

I would like to begin by thanking my parents for their continued support since I was a child. I also dearly thank Dr. Kenneth Eugene Miller for hiring me in 2008 as a research technician (while working on my Associates degree) and providing me an opportunity to work in my field of study and passion. Dr. Miller has been a mentor in many aspects of my education, and I look forward to giving others the tools he has given to me. Dr. Miller passed away from amyotrophic lateral sclerosis (ALS) a few weeks before my defense, but he prepared me the entire time while battling ALS. Due to Dr. Miller's continued help and support, I was fully prepared for graduation and feel I had an outstanding defense. With Dr. Miller's guidance, I am an author on seven scientific publications and first-author on four (of the 7). I have become a proficient scientist under the tutelage of Dr. Miller, and I deeply appreciate the opportunities I have had, and I look forward to preparing others for their scienfic journey, as he had prepared me. I would also like to thank another very special person in my life, Laurie Schreck, for her patience and help to navigate the rigors of the graduate program. Laurie provided a level of stability in my life that positioned me to surpass my own expectations. I am also very thankfuly of several faculty at Oklahoma State University Center for Health Sciences, such as Dr. Nedra Wilson, Dr. J. Thomas Curtis, and Dr. Koehler (to name just a few), who provided support that helped me prosper in the PhD graduate program.

Name: Michael Brian Anderson

Date of Degree: December, 2021

Title of Study: A NEW AGE: QUANTITATIVE MORPHOMETRY FOR THE 21ST CENTURY – ADVANCES IN THE STUDY OF DORSAL ROOT GANGLION NEURONS

Major Field: BIOMEDICAL SCIENCES

Abstract: Dorsal root ganglion (DRG) neurons transduce sensory information in peripheral tissues, transmitting centrally as action potential frequency, along second and third order neurons for processing in the somatosensory cortex. DRG neurons are the focus of study for researchers to understand pain pathways, and clinicians in the diagnosis of neuropathic disease. Adjuvant induced arthritis (AIA) is an inflammatory (arthritis) model used in pain research but apoptosis and neuronal integrity during the AIA model was unknown. Researchers study the cell body of DRG neurons to evaluate proteins and pathways involved in the maintenance of long-term pain and inflammation. Techniques for measuring the cell body of DRG neurons are accurate but take considerable time, reducing the scope of the experiment, and are susceptible to bias and error. In the diagnosis of several peripheral neuropathies, clinicians measure the cross-sectional density of protein gene product 9.5 (PGP9.5) immunoreactivity, a pan neuronal (antibody) biomarker reactive to intra-epidermal nerve fibers (IENFs); however, global two-dimensional (2D) densiometric measurement of branching three-dimensional (3D) IENFs lack resolution to evaluate individual IENFs. In these studies, I have evaluated neuronal integrity of DRG neurons at the peak time point of the AIA model and observed no apoptosis or loss of integrity. Automatic identification and segmentation of DRG neurons was achieved by developing an algorithm, in conjunction with the use of a biomarker for DRG neurons. Morphologic 3D quantitation of IENFs was achieved by epidermal isolation, tissue clearing, immunolabelling, and measurement/reconstruction, per IENF. High-throughput analysis of DRG neurons and morphometric analysis of individual IENFs opens the doors to the future of DRG neuronal analysis for both researchers and clinicians. To naturally observe, interact, and communicate 3D data, a novel use of the Unreal Engine was employed to develop an educational, and downloadable, experience.

NOMENCLATURE

2D – two-dimensions
3D – three-dimensions
A1PO – 1% acrylamide in phosphate buffer solution
A1PO/bis - 1% acrylamide with 0.0125% bis-acrylamide in phosphate buffer solution
AC – adenylate cyclase
AB - antibody
AIA – adjuvant-induced arthritis
AP – action potential
Aα – a-alpha (nerve fiber)
Aβ – a-beta (nerve fiber)
Aδ – a-delta (nerve fiber)
ALS – anterolateral system
AM – adrenomedullin
AMY – amylin
ATP – adenine triphosphate
β-actin – beta actin
BCa – bias-corrected and acceleration (bootstrap interval)
BSA – bovine serum albumin
C-II – carrageenan induced inflammation
cAMP – cyclic adenosine monophosphate
CFA – complete Freund's adjuvant
CGRP – calcitonin gene-related peptide
CGRP (-) – CGRP negative
CGRP (+) – CGRP positive
CLARITY – clear lipid-exchanged acrylamide-hybridized rigid imaging / immunostaining/ in situ-hybridization-compatible tissue hydrogel
CLR – calcitonin-like receptor
CNS – central nervous system
CT – calcitonin (family of peptides)
DNA – deoxyribonucleic acid
DAPI – 4', 6-diamino-2-phenylindole
DMEM – dulbecco's modified eagle medium
DRG – dorsal root ganglion
EDTA – ethylenediaminetetraacetic acid
EMGI – evaluation mean grey intensity (score)
FIJI – fiji is just imagej (image analysis software)
FOV – field of view
GAPHD - glyceraldehyde 3-phosphate dehydrogenase
GLU – glutamate
GLS - glutaminase
GPCR – g protein-coupled receptor

IACUC – institutional animal care and use committee
IC-DRGs – image cytometry in dorsal root ganglion neurons
IL-6 – interleukin 6 IMD – intermedin
IENF – intra-epidermal nerve fiber
IHC – immunohistochemistry
IP_3 – inositol triphosphate
IR - immunoreactivity
MAPK/ERK – mitogen-activated protein kinase
MGI – mean grey intensity
mRNA – messenger ribonucleic acid
NeuN – neuronal nuclei (protein) (also known as NeuN/FOX3)
NGF – nerve growth factor
NIH – national institutes of health
PACT – passive CLARITY Technique
PBS – phosphate buffered saline
PCML – posterior column medial-lemniscal (pathway)
PCR – polymerase chain reaction
PFA – paraformaldehyde
PIV script – pixel intensity varaiation script
PKA – protein kinase A (pathway)
PVDF – polyvinylidene fluoride
PVP – polyvinylpyrrolidone
qRT-PCR – quantitive reverse transcription polymerase chain reaction
RAMP1 – receptor activity modifying protein 1
RCP – receptor component protein
RIMS – refractive index matching solution
RNA – ribonucleic acid
RRID – research resource identifier
RT-PCR – reverse transcription polymerase chain reaction
SDS – sodium dodecyl-sulfate
SEM – standard error of the mean
SP – substance P
TrkA – tropomyosin receptor kinase A
TRPV1 – transient receptor potential cation channel, subfamily V, member 1
TRPV2 – transient receptor potential cation channel, subfamily V, member 2
TRPA1 – transient receptor potential cation channel, subfamily A, member 1
VPL – ventral posterior lateral (nucleus)

TABLE OF CONTENTS

Chapter Page

Chapter Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

Introduction

Living with Chronic Pain

Chronic pain is a continual struggle for many people around the world. There are several neuropathic diseases accompanied by chronic pain, affecting from 11% to 40% of Americans, and characterized by a reduced quality of life, depression, and worsening conditions over time (Dahlhamer et al., 2018). Injury and disease can result in the maladaptation of somatosensory neurons, disrupting normal physiologic neuronal function and inducing chronic pain and inflammation (Woolf and Ma, 2007). There is an array of techniques available to research chronic pain and diagnose peripheral neuropathies; however, due to technological limitations of current processes, the resolution is dated. With the advent of image analysis scripting, confocal microscopy, tissue clearing, volumetric measurement, and three-dimensional reconstruction arrives a quantitative revolution for the research of primary afferent dorsal root ganglion (DRG) neurons.

Sensory Transduction

Sensory information is transduced by primary afferent nerve terminal depolarization and transmitted to the cerebral cortex for perception by the somatosensory system via two functionally distinct pathways, the anterolateral system (ALS) and the posterior

column medial-lemniscal (PCML) pathway. Functionally, the ALS pathway transmits sensory modalities for pain, temperature, and crude touch, whereas the PCML pathway transmits perception for fine touch, proprioception, and vibration. Anatomically, first-order DRG neuronal axons project into the dorsal horn of the spinal cord, synapse on second-order ipsilateral neurons, decussate (cross) the midline, and extend into the ventral posterior lateral (VPL) nucleus. Whereas first order PCML DRG neuronal axons ascend dorsal columns of the spinal cord to synapse on second-order ipsilateral neurons in the medullary dorsal column nuclei, decussate in the medulla, and extend into the VPL nucleus. The path of first and second-order somatosensory neurons diverge and decussate, to synapse on third-order neurons in the VPL nucleus, and terminating in contralateral regions in the parietal lobe, providing sensory perception. Transduction of somatosensory system occurs by synaptic release of excitatory molecules, binding of compliment receptors (achieving a threshold of activation), terminal depolarization, and generation of action potential frequency.

Peripheral Neuroinflammation

Acute and chronic inflammatory pain states are coerced and maintained from nerve injury, tissue damage and metabolic imbalance, among other causes. The acute phase of inflammatory sensitization, i.e., pain, is provoked by repeated depolarization of peripheral sensory nerve terminals, causing the peripheral release of calcitonin gene-related peptide (CGRP), substance P (SP) and glutamate (GLU), and the depolarization of the DRG central pre-synaptic terminals on 2nd order neurons in the dorsal horn of the spinal cord for the release of GLU (**Figure 1.1**). After continual peripheral depolarization, neuronal soma (cell bodies) will respond to repeated

nociceptive stimuli, resulting in a somatic phenotypic alteration towards continued inflammatory sensitization, which are reflected in peripheral nerve fibers, eventually leading to the chronic phase of inflammation. Altered action potential firing patterns and retrograde transport of neurotrophins during a severe inflammatory event can lead to short- and long-term adaptations of DRG neurons (Fang et al., 2005; Djouhri et al., 2006). Inflammation caused by disease and injury can result in a malfunction of this complex cellular processing, resulting in a maladaptation of neuronal plasticity and life-altering pain disorders. The cell bodies of primary sensory neurons often are studied to understand adaptive and maladaptive alterations during immune-regulated pain (Hoffman et al., 2010).

Primary Afferent Neurons

The DRG is an encapsulated cluster of primary afferent neurons of the ALS and PCML pathways. The cell body of DRG neurons is the functional site of transcription, translation, cellular homeostasis, and genomic control. The soma (cell body) of these neurons lies near the spinal cord and project afferent nerve terminals into distant peripheral tissues, such as the hand and foot. Peripheral information must be transmitted with high fidelity along this distance to the soma, then onto second order neurons in the dorsal horn of the spinal cord before being sent to specific regions of the brain for stimulus perception. There is some conjecture to the size classification of DRG neurons in relation to functionality; while some studies describe the size population of DRG neurons as bimodal (Lawson and Biscoe, 1979; Lawson et al., 1984) others report the size distribution as trimodal (Kawamura and Dyck, 1978). Electrophysiological studies have shown that the conduction velocity of DRG neurons is related to cell size and morphology (Harper, 1985). DRG neuronal somata that project C fibers (unmyelinated) are most uniform in size and restricted to small diameter neurons, while somata with Aδ fibers (myelinated) have a slightly larger mean and range of size (Harper, 1985). The DRG neurons with C and Aδ fibers are considered to convey pain and temperature modalities. The size distribution of DRG neurons

somata with Aα and Aβ fibers (heavily myelinated) are more loosely correlated by size and conduction velocity than Aδ and C fiber neurons (Harper, 1985). Large diameter neurons that convey proprioceptive, fine touch, and vibration information, although a subset of large neurons is nociceptive (conveying pain) (Lawson et al., 1984; Djouhri et al., 2006).

DRG neuronal cell bodies receive information from peripheral nerve terminals by two primary mechanisms, action potential (AP) number/frequency and molecular cargo trafficked by dynein (retrograde transport) motor proteins. It is through these two forms of communication, electrical and physical, that peripheral information is transmitted, with high fidelity. For example, neurotrophins, such as nerve growth factor (NGF) are expressed in epidermal keratinocytes and released upon tissue injury and transported to the DRG neuronal cell body. Action potential frequency and retrogradely-transported neurotrophins initialize transcription and translation of DRG proteins required for maintaining peripheral nerve sensitivity and inflammation. Inflammatory sensitivity is maintained by somatic upregulation of sodium channels (required for depolarization and transmitting action potentials), transient receptor potential channels (TRPV1/2 and TRPA1), tropomyosin receptor kinase A (TrkA: binding NGF as retrograde complex), neuropeptides CGRP, SP and enzymes glutaminase (GLS) and aspartate aminotransferase (which both can produce GLU). Subsequently, proteins are packaged into vesicles or other organelles and physically transported peripherally via kinesin (anterograde transport) motor proteins. Biomolecules such as CGRP, SP, and GLU are released from peripheral terminals, contributing to the establishment and maintenance of chronic inflammatory conditions (**Figure 1.2**). DRG neurons are a dynamic population of cells, responsible for conveying information for pain, fine touch and innocuous temperatures, while also capable of releasing neuropeptides (such as CGRP) on peripheral epithelia during tissue injury to maintain inflammatory sensitization during the healing process. These neurons actively protect injured tissue in humans and animals, but when disease or cellular malfunction occur then this protective system may become maladapted due to ongoing alterations in the somata and IENFs. Alterations in the operational capacity of these

neurons during painful conditions are a primary interest for researchers in the pursuit of non-addictive pain management.

Intra-epidermal Sensory Nerve Fibers

Epidermal nerve fibers are the peripheral component of primary sensory neurons and allow for an organism's ability to sense the environment, i.e., proprioception, fine touch, temperature and noxious stimuli. In respect to the skin, low threshold nerve fibers are found in the vascular dermal layer and are highly myelinated Aα and β fibers, responsible for proprioception and mechanical sensation, with conduction velocities of 80-120 meters per second (m/s) and 33-75 m/s, respectively (Haines, 2002). High threshold IENFs are found in the avascular epidermis and are unmyelinated C fibers and lightly myelinated Aδ fibers, responsible for fine touch, temperature and pain, with conduction velocities of 0.5-2.0 m/s and 3-30 m/s, respectively (Haines, 2002).

Upon intense activation, IENFs and dermal nerve fibers sensitize local cells peripherally and interneurons of the spinal cord centrally, resulting in an expansion of the sensory receptive field from the original site of injury. This establishes an area of hypersensitivity that further protects the site of injury (Gangadharan, 2013; Eller-Smith et al., 2018) . In neuro-vascular tissue, injury causes the release of CGRP and SP from peripheral nerve fibers resulting in vasodilation, vasopermeation, and inflammation (**Figure 1.2**). However, the function of CGRP and their cognate receptors on IENFs and epidermal skin cells in neuro-avascular tissue is less understood. It may be that CGRP receptors are located on IENFs distinct from CGRP containing nerve fibers. For example, the localization of CGRP in rat dura mater is exclusively localized to C fibers while CLR and RAMP1 are localized to Aδ fibers, illustrating CGRP and CGRP receptor peripheral nerve specificity (Eftekhari et al., 2013). Release of CGRP, therefore, may activate other peripheral nerve fibers, recruiting previously unresponsive DRG neurons (Russell et al., 2014; Esposito et al., 2019). Due to morphological complexity, only 3D evaluation of intra-epidermal

nerve fibers is sufficient to accurately identify and evaluate individual nerves for discrete differences in CGRP and CLR/RAMP1 immunoreactivity.

The diverse range of sensory function and intracellular communication elicited from DRG neuronal primary afferent IENFs compose a highly dynamic and adaptable system that is also vulnerable to pathologic alterations. In a disease state, such as diabetes, reduced IENF density can result in a reduction of epithelial tissue, neurotrophic factors, and a degeneration of epidermal tissue. Several neuropathies are characterized by a reduction of IENF density and are frequently evaluated with two-dimensional (2D) analysis (Lauria, 2005); however, three-dimensional (3D) IENF data would provide a more complete evaluation of morphometric analysis, such as branching number, volume, surface area and IENF diameter (Anderson and Miller, 2018a).

Calcitonin Gene-Related Peptide

The calcitonin gene-related peptide (CGRP) peptide belongs to the calcitonin (CT) family of peptides, which include CGRP-α (CGRP), CGRP-β, adrenomedullin (AM), adrenomedullin 2/intermedin (AM2/IMD) and amylin (AMY) (British.Pharmacological.Society, 2009) (**Figure 1.3**). There are two types of functional G protein-coupled receptors (GPCR) for calcitonin peptides, the calcitonin receptor (CTR) and the calcitonin receptor-like receptor (CLR) (Hay et al., 2018). Functionality of either GPCR, CTR or CLR, is facilitated by one of three required receptor activity modifying proteins (RAMPs), with varying affinity to the calcitonin family of peptides (Weston et al., 2016). Additionally, the CGRP-receptor component protein (RCP) is required for CLR-RAMP1 signaling pathways (Prado et al., 2002; Karsan and Goadsby, 2015). CGRP agonism (activation) of the CLR/RAMP1/RCP receptor complex, through Gαs subunit signaling, leads to an ATP to cAMP conversion by adenylate cyclase (AC), and activation of the PKA pathway, increasing the concentration of intracellular Ca^{++} and nerve sensitivity (**Figure 1.4**) (Iyengar et al., 2017). Additionally, peripheral nerve activation by CGRP leads to nitrous oxide production and activation of the mitogen-activated protein kinase (MAPK/ERK) pathway

(Dickerson, 2013). The function of CGRP and the CGRP receptor components, CLR/RAMP1/RCP, are most commonly recognized for blood capillary vasodilation during neuro-vascular inflammation, however, the function of these receptors is less known in the neuro-avascular epidermis.

Inflammatory Models

Inflammatory models are employed to induce a cascade of inflammatory events for evaluating neuroinflammatory pathways and testing compounds. Two common models for inducing experimental peripheral inflammation are intradermal injection of carrageenan, with a peak swelling at 3-7 hours, and complete Freund's adjuvant (CFA) emulsion, with a peak swelling at 24-48 hours, persisting for several weeks (Hoffman et al., 2010; Fehrenbacher et al., 2012; Chondrex, 2017; Anderson et al., 2019). Carrageenan-induced inflammation is resolved in 24-48 hours, whereas CFA-based models, such as the adjuvant-induced arthritis (AIA) model, are selected for studies evaluating the acute and/or chronic phases of peripheral inflammation and hypersensitivity. The AIA model offers the potential to evaluate peripheral neuroinflammatory targets during antagonistic events and evaluate their effects downstream in the chronic phase of inflammation. The chronic phase of inflammation is defined as a maladaptation of normal homeostatic neuronal function (Woolf and Ma, 2007); however, the integrity of DRG neurons at the acute time-point of inflammation using the AIA model is unknown.

Traditional Methods of Peripheral and Somatic DRG Neuronal Quantification

Traditionally, image analysis of DRG soma involves the manual evaluation of individual images by hand-tracing of cytoplasmic and neuronal boundaries using a pen-input display monitor (Miller et al., 1993; Zhang, 2013; Hoffman et al., 2016). This process is effective but can take days to weeks of manual tracing and is susceptible to potentially undetected human error. There

are a minimum of 24 operations required to manually measure, and organize data, from the nuclear, cytoplasmic and whole cell compartments of one cell. The minimum number of steps dramatically increases when measuring a thousand neurons, from 24 to over 8,000 manual operations. Manual tracing is currently the standard for measuring subcellular differences in IHC processed DRG sections, however this is time consuming and error prone. Consequently, the automation of histologic image analysis has been of interest but mostly unachievable due to morphological variation and the histologic variation across different tissue sections. To overcome these issues, I have theorized a system that combines IHC and algorithmic scripting to identify a neuronal subpopulation and accurately measure cytoplasmic and nuclear compartments, within a measurable error rate.

Intra-epidermal nerve fibers (IENFs) are branching sensory networks weaving through five prominent layers of continually differentiating, interconnected and water-tight, epidermal keratinocytes. Due to complex 3D morphology, 3D measurement of IENFs is optimal, however, most IENF data is reported in only two dimensions because of processing limitations, opacity of keratinocytes, and permeability of IHC molecules. While 2D image analysis is sufficient for general IENF densitometry, only 3D image analysis has the power to accurately measure total length, volume, branching points, mean-grey-intensity (MGI) and shape per fiber. Due to these restrictions, there is currently no published data of fully reconstructed epidermal nerves in 3D from intact epidermis, illustrating a scientific limitation that must be overcome for accurate IENF measurement and evaluation. I am developing the first method for epidermal isolation (from dermis), optical clearing (for confocal imaging), and scripted algorithmic approach to IENF measurement and 3D reconstruction of individual epidermal DRG nerve fibers.

Research and Clinical Impact

In academic research, experimental questions often involve several experimental groups, resulting in large amounts of data that require processing. Experimental data in the form of micrographs

(images) require analysis for data collection. Traditional image analysis can take several weeks of manual interaction for gathering data. Several methods of automated image analysis (image cytometry) exist commercially for measuring isolated cells, such as cell/plate culture; however, image cytometry in tissue has been elusive due to heterogeneous cell populations, proximity of cells, morphological variability, available biomarkers, and nonspecific fluorescent artifacts. Manual image analysis of DRG neurons can take up to eight hours per group, presenting as a temporal bottleneck in the technique, which translates into fewer numbers (samples) per experiment.

In the clinical setting, several peripheral neuropathies, such as diabetes, are characterized by a reduced IENF density and can result in a reduction of epithelial tissue neurotrophic factors and a degeneration of epidermal tissue. Traditionally, whole skin (epidermal/dermal layers) is processed with immunohistochemistry (IHC), cross-sectioned and evaluated for IENF density, a technique commonly performed by clinicians in the diagnosis of several peripheral neuropathies (Lauria, 2005). However, the nature of sprawling 3D structures, weaving in and out of a single 2D focal plane, can lead to incomplete IENFs for measurement. IENF fragmentation in 2D images restrict the discrete morphologic measurement of neuronal populations, volume, branching points, and protein content, among other 3D measurements, per nerve. Therefore, a comprehensive method of intra-epidermal nerve fiber volumetric and morphological evaluation has the potential to revolutionize the clinical diagnosis of several peripheral neuropathies.

Summary

Chronic pain and inflammation are debilitating and life changing symptoms of peripheral neuropathic diseases, suffered by more than an estimated 20 million people in the United States (Johannes et al., 2010; Kennedy et al., 2014; Dahlhamer et al., 2018; Stroke, 2021). Primary somatosensory neurons in the DRG are responsible for transducing environmental and visceral

sensory pain signals, deriving pain and inducing protective behavior at the injury site. Severe tissue damage and disease can induce neuronal injury, coercing a maladaptation during neuronal recovery, altering homeostatic function, and resulting in symptoms of chronic pain, inflammation, and long-term suffering. Academic research and clinical diagnosis tools for measuring pain and inflammation rely on accurate but outdated technologies and are a prime focus for modernization with current and evolving technologies, to increase efficiency and accuracy. The AIA model in the hind paw of the rat is a common technique used by academic pain researchers to emulate maladapted neuronal function in the chronic phase of inflammation; however, ***the integrity of DRG neurons and their ability robustly respond at a peak time-point of peripheral swelling is unknown (Chapter 2)***.

In neuroinflammatory pain research, primary afferent neurons of the dorsal root ganglion (DRG) are studied to understand molecular signaling mechanisms involved in nociception (pain) and inflammation. Measuring IHC (immunofluorescence) in DRG neurons requires manual hand tracing of nuclear and somatic boundaries, which is laborious, error-prone, and may require several weeks to collect the appropriate sample size with a mouse or pen-input display monitor. Automated image analysis is a power advance by coding specific functions that can be re-ran in future studies and shared for optimizing compatibility between collaborating labs around the world. Automation of image analysis (image cytometry) is a common practice in techniques involving data collection in images captured from isolated cells (cell culture), due to a strong contrast between negative and positive immunofluorescence, however, elusive in tissue sectioned slides imaged in two-dimensional cross-sections. ***An efficient and novel technique for automating data collection in tissue-processed and fluorescently labeled two-dimensional images would relieve a temporal bottleneck in somatosensory research, reducing error, increasing compatibility, and standardizing data collection (Chapter 3).***

The measurement of intra-epidermal nerve density is a powerful tool in the diagnosis and progression of several peripheral neuropathies. Due to logistical difficulty in processing,

traditional and current two-dimensional methods of measuring IENF density have poor resolution. However, a wide variety of nerve terminals reside in the epidermis with a variety of morphologies, constitution pf proteins, and sensory modalities. ***Through modern advances in most every scientific field, a juxtaposition of biomedicine, software coding, tissue clearing biochemistry, and virtual three-dimensional rendering could reveal the secret and subtle intricacies of intra-epidermal nerves, in healthy and through the progression peripheral neuropathic states (Chapter 4).***

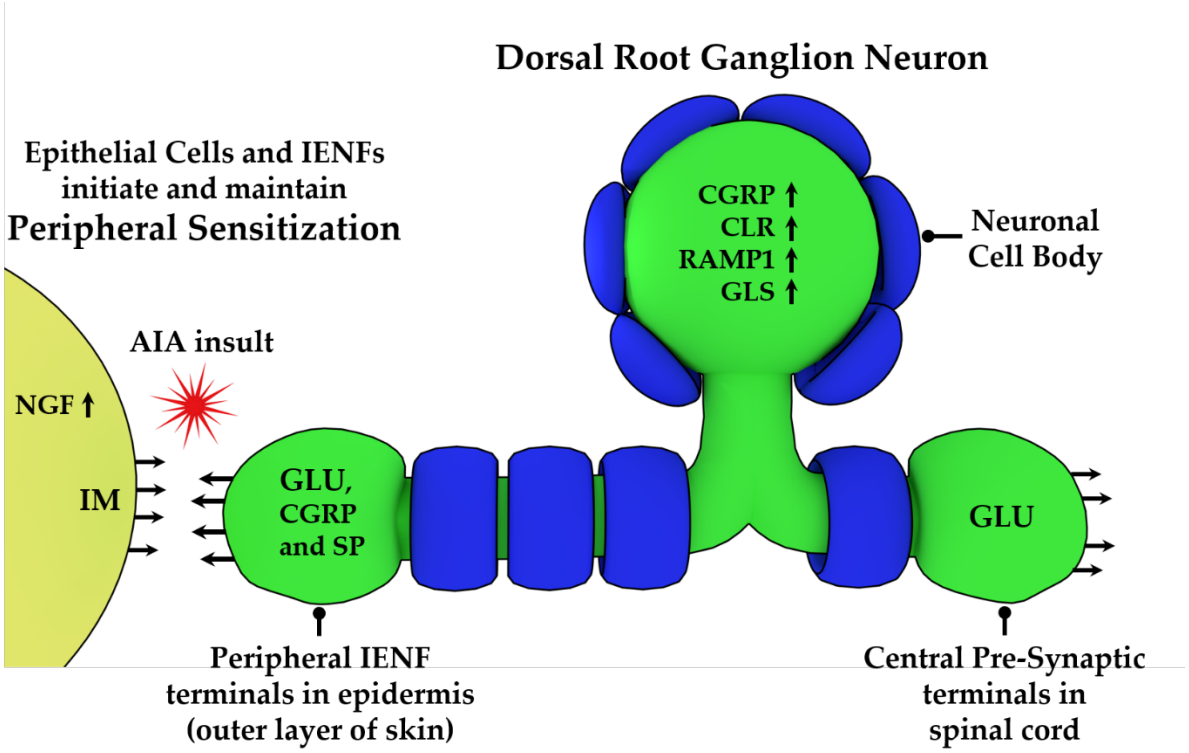


Figure 1.1 The Primary Afferent DRG Neuron. The DRG neuron: upon nerve terminal depolarization, calcitonin gene-related peptide alpha (CGRPα), Substance P (SP) and glutamate (GLU) are released from intra-epidermal nerve fibers (IENFs) to initiate and modulate inflammatory sensitization. In response, epithelial cells release inflammatory mediators (IM) and nerve growth factor beta (NGF-β), initiating sensitization. Centrally, GLU is released in the dorsal horn of the spinal cord. In the DRG neuronal cell body, CGRP, calcitonin receptor-like receptor (CLR), receptor activity modifying protein 1 (RAMP1) and glutaminase (GLS) are increased to maintain the chronic inflammatory phase, somatically. The blue structures surrounding the neuronal cell body and axons are satellite cells and Schwann cells, respectively.

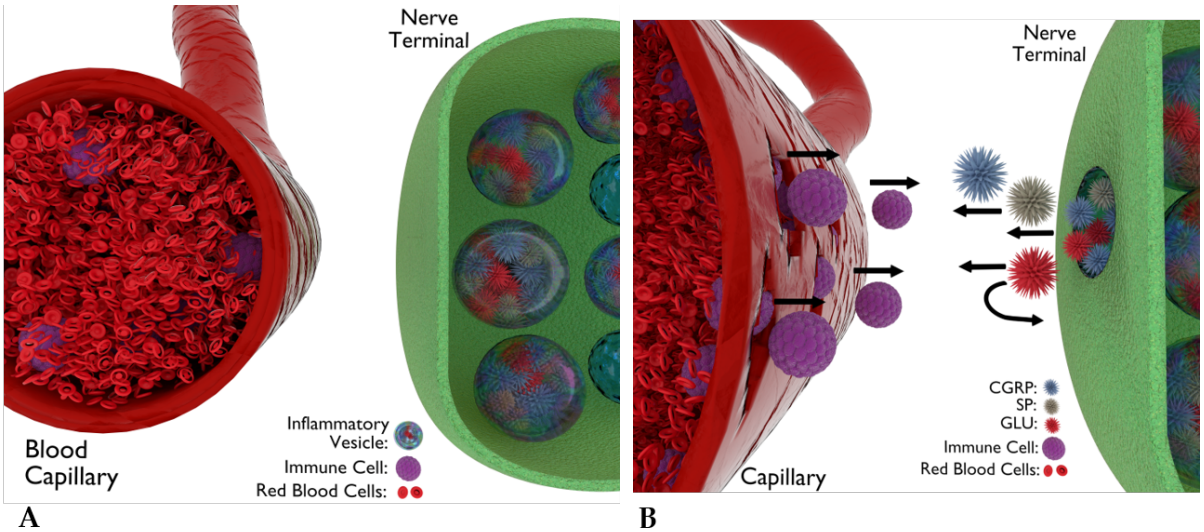


Figure 1.2 Peripheral release of CGRP. In response to AIA insult, peripheral nerve fibers release of CGRP and SP, leading to vasodilation and permeabilization of local blood capillaries. A, In normal conditions, nerve terminals do not release CGRP and local blood vessels do not dilate or become permeable. B, in response to peripheral insult by the AIA model, CGRP, SP and GLU are released from sensory nerves. CGRP dimerizes with CGRP receptors on blood vessels, provoking neutrophil/plasma extravasation and establishing the acute phase of inflammation.

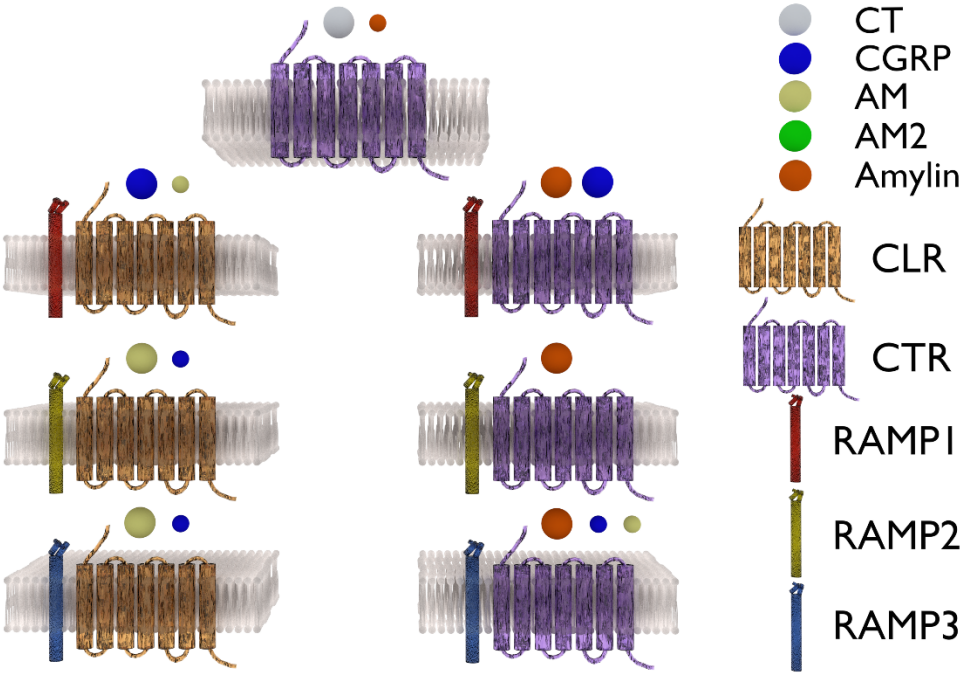


Figure 1.3 CGRP Receptor Components and Ligand Affinity. The calcitonin (CT) family of peptides include α-CGRP (CGRP), β-CGRP, adrenomedullin (AM), adrenomedullin 2/intermedin (AM2/IMD) and amylin (AMY). One of two forms of the CGRP receptor, the calcitonin receptor (CTR) and calcitonin receptor-like receptor (CLR), and one of the three receptor activity modifying proteins (RAMPS I – III) are required for CGRP receptor functionality.

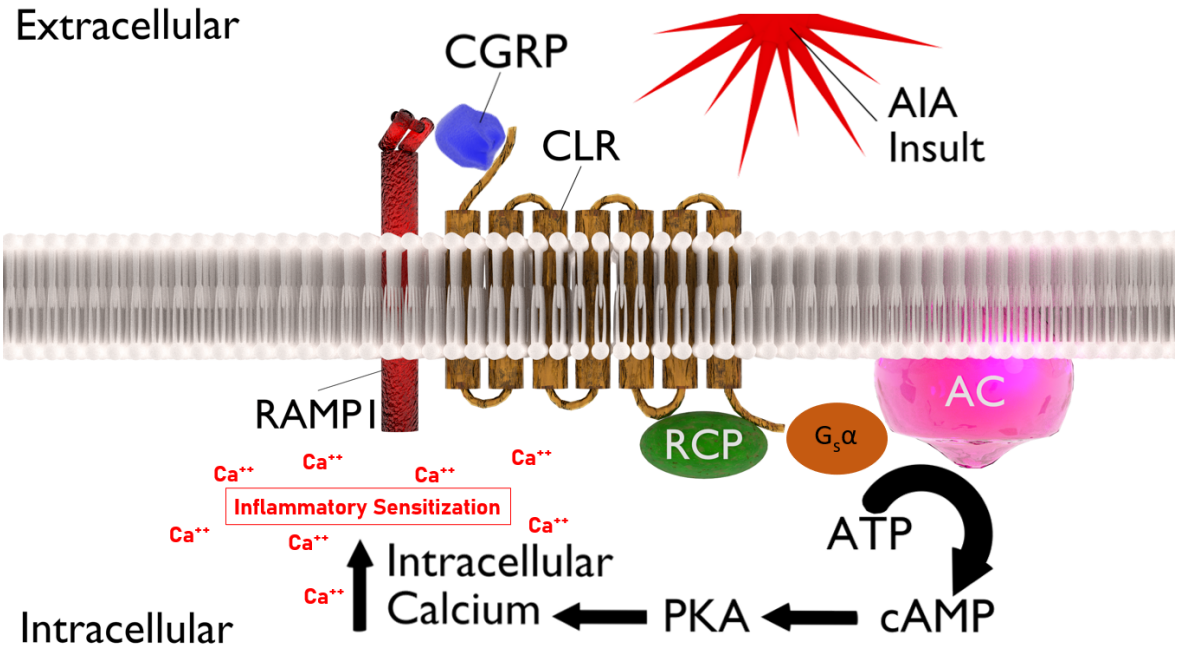


Figure 1.4 The CLR/RAMP1/RCP receptor complex. In response to peripheral AIA insult, CGRP is released from IENFs and dimerizes to the CLR/RAMP1/RCP complex, activating the CGRP receptor. RCP facilitates the signaling and exchange of guanine diphosphate (GDP) with guanine triphosphate (GTP) on G protein alpha (Gα) subunits, initiating adenylate cyclase to convert ATP to cyclic-AMP (cAMP). With increased intracellular cAMP production, protein kinase A (PKA) is activated, increasing intracellular calcium, cellular excitability and contributing to peripheral sensitivity. Activation of the CGRP receptor in the absence of RCP signaling may initiate secondary pathways, capable of nitrous oxide production, potassium-ATP channels and activation of the MAPK/ERK pathway in the modulation of peripheral sensitization.

CHAPTER II

Subcellular localization of neuronal nuclei (NeuN) antigen in size and calcitonin gene-related (CGRP) populations of dorsal root ganglion neurons during acute peripheral inflammation

Abstract

Pseudo-unipolar cell bodies of somatosensory primary neurons are located in the dorsal root ganglia (DRG). The somatic and peripheral domains of DRG neurons are often studied in sensory pain research to understand molecular mechanisms involved in the activation of pain and maintenance of inflammation. Adjuvant-induced arthritis (AIA) is an inflammatory model that elicits a robust and rapid onset immune response with a maximal swelling period of 24-48 hours and persisting for several weeks. The AIA model in the hind paw of the rat elicits a potent inflammatory response of the dermis and epidermis, leading to protein expression changes for sensitization of many DRG neurons; however, it is unknown if the AIA model in the hind paw of the rat induces DRG neuronal injury, necrosis, or apoptosis at the somatic level. Neuronal nuclei (NeuN) antigen is a biomarker for post-mitotic neurons, neuronal identification, protein alterations, injury, and loss.

Calcitonin gene-related peptide (CGRP) is expressed in C and Aδ DRG neurons, a subset of DRG neurons known to play a role in peripheral sensitization. The focus of this research was to evaluate the expression pattern of NeuN immunoreactivity, in size (soma) and CGRP subpopulations of DRG neurons in naïve and inflamed groups. Confirmed by both immunofluorescence and immunoprecipitation, DRG neuronal expression of NeuN was localized to nuclear and cytoplasmic subcellular compartments. NeuN increased within the nucleus of small CGRP positive DRG neurons during inflammation, indicating a potential role for NeuN in a subset of nociceptive neurons.

Introduction

Primary sensory neurons of the somatosensory pathway project axons from the dorsal root (or spinal) ganglia (DRG) over great distances and terminate in peripheral tissue as multiple branched afferent nerve terminals. The complexity of pseudo-unipolar DRG neurons include four main functional domains: the peripheral terminal (environmental sensory activation), the DRG soma (gene and protein response), information conveyance (axon action potential), and the relay of information to the central nervous system (synaptic transmission). Altered action potential firing patterns and retrograde transport of neurotrophins during a severe inflammatory event can lead to short- and long-term adaptations of DRG neurons (Fang et al., 2005; Djouhri et al., 2006). Inflammation caused by disease and injury can result in a malfunction of this complex cellular processing, resulting in a maladaptation of neuronal plasticity and life-altering pain disorders.

The cell bodies of primary sensory neurons often are studied to understand adaptive and maladaptive alterations during immune-regulated pain (Hoffman et al., 2010). A common model for inducing peripheral inflammation is intradermal injection of complete Freund's adjuvant (CFA), resulting in a peak swelling at 24-48 hours and persisting for several weeks (Hoffman et

al., 2010). The AIA model in the hind paw of the rat induces peripheral inflammation with increased mechanical sensitivity and a peak inflammation of 48-hours (Hoffman et al., 2016).

I and others have shown that DRG neuronal cell bodies upregulate the production of pro-sensitization proteins in the course of the CFA model (Hoffman et al., 2010; Hoffman et al., 2011; Qu and Caterina, 2016). During CFA-induced inflammation in the rat hind paw, dermal axons and intra-epidermal nerve fibers (IENFs) from the DRG are located within the inflammatory edema; however, it is unknown if the DRG neuronal cell body becomes injured, necrotized, or apoptotic during the peak swelling event.

To evaluate neuronal plasticity, injury, or loss, I chose Neuronal Nuclei (NeuN), a neuronal biomarker involved in transcriptional regulation. The NeuN antigen is a 106 amino-acid epitope mapped to the N-terminal of forkhead box (FOX)-3 protein (Kim et al., 2009) and is expressed in the nuclei of post-mitotic neurons (Mullen et al., 1992). The function of NeuN/FOX-3 is a post-mitotic splicing regulator using cell-specific alternative splicing in the nuclei of most neurons (Kim et al., 2009; Duan et al., 2016). While the NeuN antigen was originally reported exclusive to neuronal nuclei, it has since been determined to be in neuronal cytoplasm, both, or completely absent (Van Nassauw et al., 2005; Gusel'nikova and Korzhevskiy, 2015; Ma et al., 2016). NeuN/FOX-3 expression is variable, including increasing, decreasing, or not changing, in different experimental models. In rat hippocampal neurons (using a telomerase reverse transcriptase protein) and mouse motor neurons (cytoplasmic increase in aging model), NeuN/FOX-3 expression can increase (Ma et al., 2016; Reynolds et al., 2016; Baruch-Eliyahu et al., 2019). A loss of NeuN antigenicity occurs in certain neuronal injury models, while neurons appear to be functioning, with a reappearance of NeuN/FOX-3 immunoreactivity (IR) at later timepoints (McPhail et al., 2004; Unal-Cevik et al., 2004). Alternatively, models causing neuronal injury or neuronal death often show a decrease in NeuN/FOX-3 expression (McPhail et al., 2004; Collombet et al., 2006; Wu et al., 2010; Sajja et al., 2014).

CGRP is a potent vasodilator, expressed in C and Aδ neurons, and functions in the transmission of nociceptive signaling and inflammatory wound healing (Iyengar et al., 2017). While CGRP positive (+) and CGRP negative (-) DRG neurons both play a role in nociception and peripheral sensitization, they are functionally different (Breese et al., 2005). Evaluation of CGRP in DRG neurons was performed in the present study to examine subsets of DRG neurons involved in the acute peripheral sensitization at a peak AIA time point. AIA was induced in the rat hind paw and NeuN/FOX-3 IR in the nucleus and cytoplasm of L4 DRG neurons was evaluated in CGRP (+) and CGRP (-) subpopulations. Nuclear and cytoplasmic subfractionation of DRG neurons followed by NeuN/FOX-3 immunoblotting was used to verify immunohistochemical data. To further investigate the integrity of DRG neuronal nuclei in the AIA model, 4′,6-diamidino-2-phenylindole (DAPI) fluorescence was measured for area, perimeter and mean grey intensity (MGI). I hypothesize that all DRG neurons express NeuN/FOX-3 while retaining neuronal viability at the peak swelling point of the AIA model. I expect an increase in NeuN/FOX-3 expression due to an increased demand of RNA splicing during inflammatory conditions. However, if DRG neuronal injury or loss occurs, NeuN/FOX-3 was expected to decrease, indicating a reduction of transcription in some DRG neurons.

Methods

Animals

Male and female Sprague Dawley rats (Charles River; n=21 150-250 grams; 8 weeks of age) were bred, maintained on-site, and used in IHC and Western blot experiments. All rats were given food and water *ad libitum* and housed with a 12-hour on/off light cycle. All methods and techniques utilized in these experiments were conducted in accordance with the National Institute of Health (NIH, https://www.ncbi.nlm.nih.gov/books/NBK43327/), the International Association for the Study of Pain (IASP, https://www.ncbi.nlm.nih.gov/pubmed/6877845) and approved by

the Oklahoma State University Center for Health Sciences Institutional Animal Care and Use Committee (2016-03). All efforts were made to care for and minimize the total number of rats used in these experiments to reduce potential suffering.

48-hour Adjuvant-Induced Arthritis Model

Complete Freund's adjuvant (CFA) was emulsified in a 1:1 ratio with phosphate buffered saline (PBS) and injected into the right hind paw of experimental animals to establish unilateral adjuvant-induced arthritis. For IHC (n=9) and Western blot analysis (n=12), all rats (n=21) were housed for 48-hours with food and water provided *ad libitum*. The AIA group were anesthetized with isoflurane (1.5 liters'/minute oxygen, 2% isoflurane) and injected with 150 µl emulsified CFA in the center of the right hind paw with a 26-gauge needle and allowed to recover. The metatarsal region of the right hind paw of all animals was measured with a dial caliper shortly before transcardial perfusion (Hoffman et al., 2010).

Immunohistochemistry

After 48 hours, rats (n=9) were deeply anesthetized with Avertin, a tribromoethanol anesthesia, diluted at 2.5% (*v/v*) in PBS (pH: 7.3) and administered intraperitoneally (IP) (Hoffman et al., 2010; Wang and Miller, 2016). To determine level of anesthesia, a three-point flinch test was performed on the eye, tail, and hind paw. Once rats were determined unresponsive, 1.0 ml xylazine (100mg/ml) was delivered IP and approximately 1 minute later rats were transcardially perfused with 100 ml calcium-free tyrodes (pH: 7.3), followed by fixative. Transcardial perfusions were performed with a peristaltic pump at a rate of 37 ml/minute and a total volume of 425 ml of fixative, per rat. Rats were perfused with a modified Zamboni's fixative: 0.75% (*w/v*) picric acid, 0.2% (*w/v*) paraformaldehyde (PFA), phosphate buffered saline (PBS, pH: 7.3) (Hoffman et al., 2010). This low aldehyde fixative has been shown to provide optimal labelling in DRG neurons (Hoffman et al., 2010). The right ipsilateral lumbar 4 DRG were collected, and

post-fixed for 3 hours at 4°C and transferred into PBS with 10% (*w/v,* pH 7.3) sucrose, overnight (Lin et al., 2011; da Silva Serra et al., 2016). DRGs were placed into a single mold with M1 embedding matrix and frozen with liquid nitrogen. Frozen DRG sections (12 µm) were cut on a Leica cryostat and thaw-mounted on gelatin-coated glass slides. All slides were dried for 60 minutes on a slide warmer (37°C) before antibody incubations and PBS washing steps. Primary antibody was diluted in PBS with 0.5% (*w/v*) bovine serum albumin (BSA), 0.5% (*w/v*) polyvinylpyrollidone (PVP), and 0.3% (*v/v*) Triton-X100, pH: 7.3. Chicken polyclonal NeuN antisera (Millipore, ABN91, RRID:AB_11205760) was diluted to 0.5 µg/ml (1:1000). Mouse monoclonal antibody for CGRP was diluted to 0.1 µg/ml (1:2000) (Santa Cruz, 57053, RRID:AB_2259462) (He et al., 2019). Nuclear staining was achieved by using DAPI at a concentration of 300 nM (Thermo Fisher Scientific D3571, RRID:AB_2307445). Antibodies raised against NeuN and CGRP were combined for multiplex labeling in PBS-BSA-PVP-Triton-X (pH, 7.3). Slide containers were sealed with parafilm to prevent evaporation and placed on a rocker at 4°C for 96 hours. All slides were washed three times in PBS for 10 minutes each at room temperature to remove unbound antisera. Tissue was incubated in secondary antisera with donkey anti-chicken 488 (Jackson Immuno Research Labs, 703-545-155, RRID:AB_2340375) and donkey anti-mouse Alexa Fluor 647 (Thermo Fisher Scientific, A-31571, RRID:AB_162542). Secondary antibodies were diluted to 1 µg/ml (1:1000) in PBS with 0.3% (*v/v*) Triton-X100, combined for multiplex labeling, and incubated on a rocker for one hour at room temperature. During secondary incubation, and subsequent incubation steps, all slide containers were covered with aluminum foil to prevent fluorophore bleaching from ambient light. Slides were washed three times in PBS, incubated in 300nM DAPI for 15 minutes, and washed three times in PBS before cover-slipping in Prolong Gold anti-fade mounting media.

Imaging

Images were collected on a BX51TRF Olympus epifluorescence microscope with a SPOT-RT470 camera at a pixel resolution of 1024x1024, and saved in *.tiff format. Camera exposures used for experimental evaluation were determined before image acquisition for data collection and unchanged throughout the imaging process. One image per channel was taken, per field of view (FOV), with an Olympus UPlanFL 20x/0.50, ∞0.17 objective for NeuN (exposure: 250 milliseconds (ms), gain: 2), CGRP (exposure: 2.5 seconds, gain: 2), and DAPI (exposure: 400 ms, gain: 2). Sampling of the FOVs were randomly selected for a high density of DRG neurons and collected from different slides and sections. As in our previous studies, a threshold for differentiating CGRP (+) from CGRP (-) DRG neurons was determined by the frequency distribution and smooth spline fitting of CGRP IR in all naïve DRG neurons (Hoffman et al., 2010; Hoffman et al., 2011; Wang and Miller, 2016)

Experimental Design: Manual Tracing

All images were manually hand traced in ImageJ version 1.48v (Schneider et al., 2012), using a Wacom Cintiq 21UX monitor. DRG neuronal size is related to action potential conduction velocity and is loosely correlated with function; accordingly, I measured and categorized all neurons as small (<400 μm^2), medium (400 – 800 μm^2), or large (>800 μm^2) (Harper, 1985). The ascending order of CGRP IR was evaluated by frequency distribution and then smooth spline fitting, with 4 knots, to determine the threshold for CGRP (8-bit range of CGRP: 39-255) for classifying DRG neurons as CGRP (+) or CGRP (-) (Schou et al., 2017).

Subcellular Fractionation

Right L4 and L5 DRGs were dissected from each rat (n=8) (Lin et al., 2011). DRGs were immediately minced and placed in 1.5 micro-centrifuge tubes with 200 µl STM buffer and 2 µl protease inhibitor, then processed using a previously reported technique (Dimauro et al., 2012).

Immunoblot analysis

Subfractionated samples were measured with a BCA assay kit to standardize protein concentration**.** All samples were processed for electrophoresis by adding 10 µl loading dye (0.25 mmol/L Tris, 50% glycerol, 2% sodium dodecyl sulfate (SDS), 0.01% bromophenol blue, and 50 µl/ml β-mercaptoethanol, pH: 6.8) to 40 µg protein, per sample, and boiled at 100°C, for 10 minutes. Purified protein was loaded at a concentration of 40 µg total protein per lane and processed with Bio-Rad 7.5% TGX acrylamide kit (Bio-Rad, 1610171). Electrophoresis was performed in running buffer (191.8 mmol/L glycine, 24.8 mmol/L Tris, and 3.5 mmol/L SDS) at 100 V for 15 minutes and 150 V for 40 minutes. Proteins were transferred onto nitrocellulose membranes, in transfer buffer (191.8 mmol/L glycine, 24.8 mmol/L Tris, and 20% methanol), and incubated in PBS-tween with 5% BSA. PBS with 0.3% tween was used in all steps. Chicken polyclonal NeuN antisera was diluted 0.5 µg/ml (1:1000). Mouse anti-lamin B (ThermoFisherScientific, 33-2000, RRID:AB_2533106) normalized nuclear signals and was diluted 2.5 µg/ml (1:200) (Invitrogen, 33-2000, RRID:AB_2533106). Rabbit anti-GAPDH (Cell Signaling Technology, 2118S, RRID:AB_561053) normalized cytoplasmic signals and was diluted 0.014 µg/ml (1:3000). Primary antibodies were visualized by donkey anti-chicken 488 (Jackson Immuno Research Labs, 703-545-155, RRID:AB_2340375) and donkey anti-mouse Alexa Fluor 647 (Thermo Fisher Scientific, A-31571, RRID:AB_162542) secondary antibodies, diluted to 1 µg/ml (1:1000). Immunoblots were imaged on a Typhoon Scanner 9410 and all

channels were visualized at 500 V. Data for NeuN/FOX-3 protein concentration was normalized by naïve/control (CTR) lanes and analyzed in Graphpad Prism 8.

Statistical Analysis

Power analysis (α=0.05, power=0.80, mean 1=100, mean 2=120) indicated a sample size of n=4 for each of our groups (https://clincalc.com/stats/samplesize.aspx), therefore n=4 for naïve and n=4 for AIA, per IHC and WB techniques. Evaluation of NeuN/FOX-3 IR, hind paw edema, and Western blots were processed in Graphpad Prism 8. Hind paw edema was quantified by comparing the metatarsal means from naive CTR (n=4) and AIA (n=4) conditions. The grand mean of NeuN/FOX-3 IR between naïve and AIA groups were compared with the Mann-Whitney test. Significance was determined when the p-value was less than 0.05. Normality of the grand mean (CTR, n=4; AIA, n=4), for one-way ANOVA analysis, of NeuN/FOX-3 IR was analyzed by the Shapiro-Wilk test. Multiple comparisons were statistically evaluated by parametric one-way ANOVA with Tukey post-hoc analysis. Significance was determined when the p-value was less than 0.05. Cytoplasmic and nuclear lanes of Western blots were normalized by GAPDH and lamin B lanes, respectively, and plotted in Graphpad Prism 8. Data from the cytoplasm of naïve rat three was removed due to a faint, although visibly present, GAPDH band. Western blot data were analyzed for normality by the Shapirio-Wilk test and statistically evaluated by parametric Student's t-test. Significance was determined when the p-value was less than 0.05.

Results

Hind Paw Edema

There was a significant difference (p=0.0286) in metatarsal thickness between naive (IHC, n=4, mean=3.45 mm, standard deviation (SD)=0.5492; WB, n=4, mean=3.363 mm, SD=0.3301) and AIA (IHC, n=4, mean=8.11 mm, SD=0.2839; WB, n=4, mean=7.5 mm, SD=0.2839) rats for IHC

and for subcellular fractionation/WB experiments, respectively. The increase of hind paw thickness for rats in IHC experiments was 2.35-fold and 2.23-fold in WB experiments.

Imaging

All images were collected from 3-5 slides, 3-5 sections per L4 DRG, yielding 3-5 total FOVs per rat. There were at least 135 neurons evaluated per group, except for rat 7 from the AIA group (**Table 1.1**). Many FOVs from Rat 7 contained mostly axons, resulting in fewer evaluated sections and measured neurons than other groups (**Table 1.1**). Evaluation of CGRP IR in naïve rats by frequency distribution and smooth spline fitting, with 4 knots, indicated that an 8-bit mean grey intensity value of 39 as a reliable threshold, differentiating CGRP (+) from CGRP (-) DRG neurons (**Figure 2.1**).

NeuN/FOX-3 expression in DRG neurons

In both naïve and AIA conditions, NeuN IR MGI was relatively consistent in total and CGRP sub-populations, except for nuclei of CGRP (+) neurons (**Table 1.1, 2.1; Figure 2.1**). In CGRP (+) neurons, NeuN IR MGI was greater in the cytoplasm and nuclei of medium- and large-sized neurons than small neurons (**Table 1.1; Figure 2.2**. In AIA rats, NeuN IR MGI was larger in both the cytoplasm and nucleus of large CGRP (+) neurons than small CGRP (+) neurons (**Table 1.1; Figure 2.2**). Furthermore, no statistical differences were found within CGRP (-) and CGRP (+) neurons, despite CFA-induced peripheral inflammation. (**Table 1.2; Figure 2.3**). Nuclear NeuN IR increased significantly, from naïve to AIA conditions, by 27.3% in small CGRP (+) nuclei, whereas there was no increase in the cytoplasm (**Figure 2.4**). There was no significant difference in the average number of CGRP (+) neurons between naïve and AIA groups. Qualitatively, DRG neurons from the naïve group exhibited a larger range of IR (from

light to intensely labeled) of NeuN/FOX-3 IR within the same size populations than DRG neurons from the inflammatory group (intensely labeled) (**Figure 2.5, A, E**). Furthermore, many neurons in the AIA group which were positive for CGRP were also intensely labeled for NeuN/FOX-3 IR (**Figure 2.5, D, H**).

DAPI labeling in DRG Neuronal Nuclei

Previous reports have shown that a reduction of nuclear area and nuclear perimeter, defined by DAPI, is observed in apoptotic cell nuclei (Mandelkow et al., 2017). To evaluate neuronal integrity, I measured DAPI labeled DRG neuronal nuclei between naïve and AIA experimental conditions for DRG neuronal nuclei for area, perimeter and MGI. There were no significant differences of DRG neuronal nuclear DAPI MGI, area, or perimeter between any groups.

Western Blots

Anti-NeuN labels Fox-3 gene products as a doublet, typically from 45-50 kDa, and often a second doublet at ~70 kDa. The first doublet (45-50 kDa) are the NeuN antigens described by Mullen *et al* and the second doublet at 70 kDa was identified by Kim *et al* as primarily synapsin I (Mullen et al., 1992; Kim et al., 2009). Two NeuN/FOX-3 doublets were observed at 46/48~ kDa and 64-66 kDa (**Figure 2.6, A, C**). The cytoplasmic control, GAPDH, was exclusively present in the cytoplasmic lanes at 40 kDa (**Figure 2.6, A**). The nuclear control, lamin B, resulted in a specific band at 70 kDa in the nuclear lanes (**Figure 2.6, C)**. The relative expression of NeuN/FOX-3 protein in cytoplasmic and nuclear CTR and AIA lanes determined for normal distribution and statistically evaluated by parametric t-test, resulted in no significance, respectively ($p= 0.5521$, $p = 0.0752$, **Figure 2.6, B, D**).

Discussion

The repeated activation of DRG nerve terminals during inflammation coerces DRG somatic upregulation of CGRP and several pro-inflammatory (~6), pro-nociceptive (~16), and pro-apoptotic (~8) factors (Ji et al., 2002; Iyengar et al., 2017; Martin et al., 2019), a genomic symphony of mRNA transcription and protein translation. NeuN/FOX-3 is a splicing regulator, a protein involved in transcription, and thus I expected it to increase as neurons respond to peripheral inflammation. NeuN/FOX-3 expression is elevated in some central neurons, such as rat hippocampal neurons when stimulated with inositol triphosphate (IP_3) and sigma receptor antagonists and telomerase increasing compounds [6, 38]. NeuN/FOX-3 levels also rise in the cytoplasm of motor neurons in G93A SOD1 mice as the animals age [28]. In the current study, an increase of NeuN/FOX-3 IR occurred in DRG neuronal nuclei of small diameter neurons, a size population of DRG neurons responsive to inflammation. This is an indication that a subset of nociceptive neurons is responding to peripheral inflammation through a NeuN/FOX-3 mechanism. NeuN/FOX-3 regulates pre-mRNA splicing by interacting with protein associated splicing factor (PASF) and binding to the UGCAUG RNA element (Kim et al., 2011). Future studies should involve evaluation of the specific RNAs regulated by NeuN/FOX-3 and PASF in response to peripheral inflammation.

I also considered that AIA might decrease NeuN/FOX-3 in some DRG neurons. The AIA model produces a disruptive blister in the epidermis and dermis, but it is unknown whether potential peripheral loss of axon and terminal DRG neuronal domains in the blister region results in a loss of neuronal integrity or apoptosis. Models causing neuronal injury or neuronal death often show a decrease in NeuN/FOX-3 expression, such as: peripheral (facial) nerve crush rebounding at 28 days (McPhail et al., 2004; Unal-Cevik et al., 2004), soman poisoning by 24-hours (Collombet et al., 2006), blast pressure (Sajja et al., 2014), irradiation (Wu et al., 2010), and different ischemic models (Davoli et al., 2002; Unal-Cevik et al., 2004; Portiansky et al., 2006). In the central fluid

percussion injury model in rats, however, NeuN/FOX-3 did not change at any timepoint in the lateral neocortex (Hernandez et al., 2019). A loss of NeuN immunoreactivity occurs in certain neuronal assault models, while neurons appear to be functioning, with a later reappearance of NeuN/FOX-3 IR after 2-3 weeks (McPhail et al., 2004; Unal-Cevik et al., 2004). Based on these studies and considering its post-mitotic functionality, NeuN/FOX-3 has become known as a reference for neuronal viability and possibly a surrogate marker for irreversible nerve injury, but not as a strict marker for apoptosis (Unal-Cevik et al., 2004; Alekseeva et al., 2015; Duan et al., 2016). In the current study, there was no decrease in observed NeuN/FOX-3 IR with either immunohistochemistry or subcellular fractionation and western blotting. With DAPI fluorescence, there were no signs of chromatin condensation or DNA fragmentation. These data, therefore, indicate there are no signs of DRG neuronal injury at the 48-hour timepoint of the AIA model.

In summary, CFA or the inflammation skin blister do not appear to cause a severe injury of DRG neurons at 48-hours of AIA. Assessing NeuN/FOX-3 or other injury related proteins at additional AIA time points or peripheral injury models may shed light on the reaction of DRG neurons to peripheral trauma. NeuN/FOX-3, however, did increase in small CGRP (+) neurons indicating a unique role for this transcription regulator in a subset of nociceptive neurons. Evaluating the mechanism of action for NeuN/FOX-3 during acute inflammation may provide a new therapeutic target for controlling pain.

Published 2021

Neuroscience Letters. Volume 760:135974, June 2021

https://pubmed.ncbi.nlm.nih.gov/34146639

doi: 10.1016/j.neulet.2021.135974

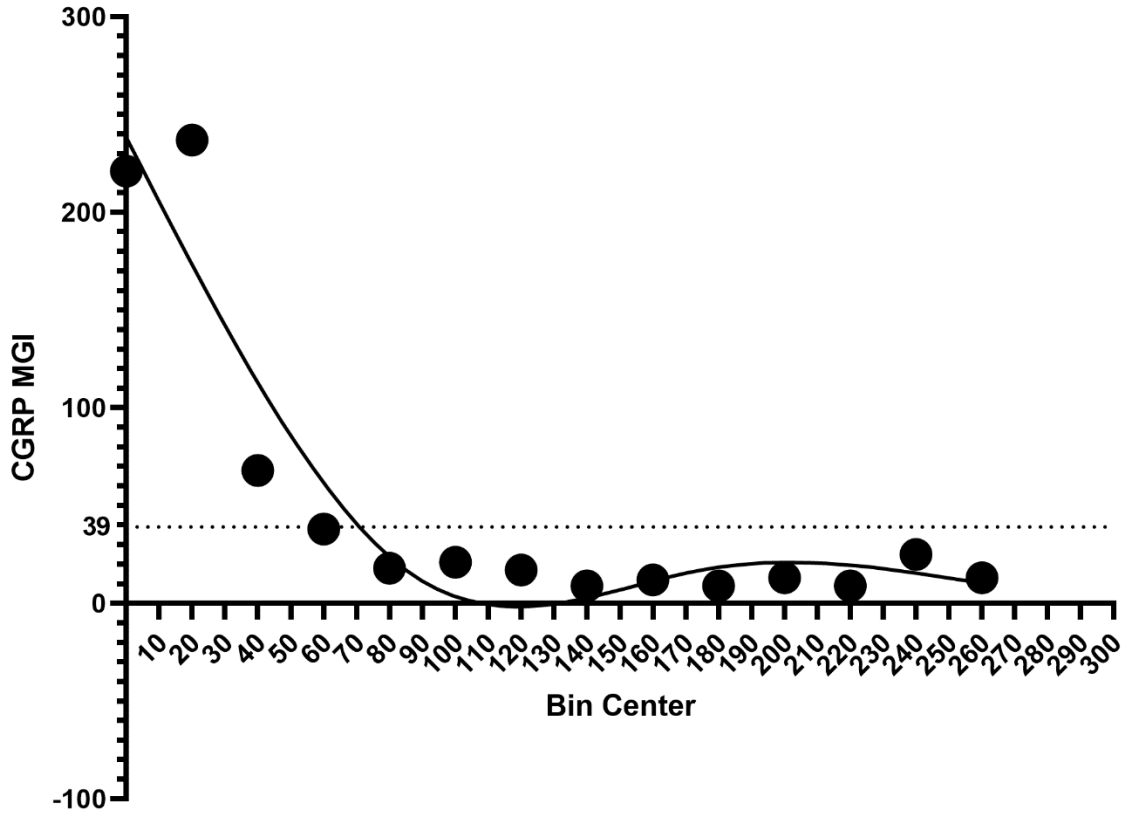


Figure 2.1 Frequency distribution of CGRP IR in naïve rats. Evaluation of CGRP IR in naïve rats by frequency distribution and smooth spline fitting, with 4 knots, indicated that an 8-bit mean grey intensity (0-255) value of 39 as a reliable threshold, differentiating CGRP (+) from CGRP (-) DRG neurons.

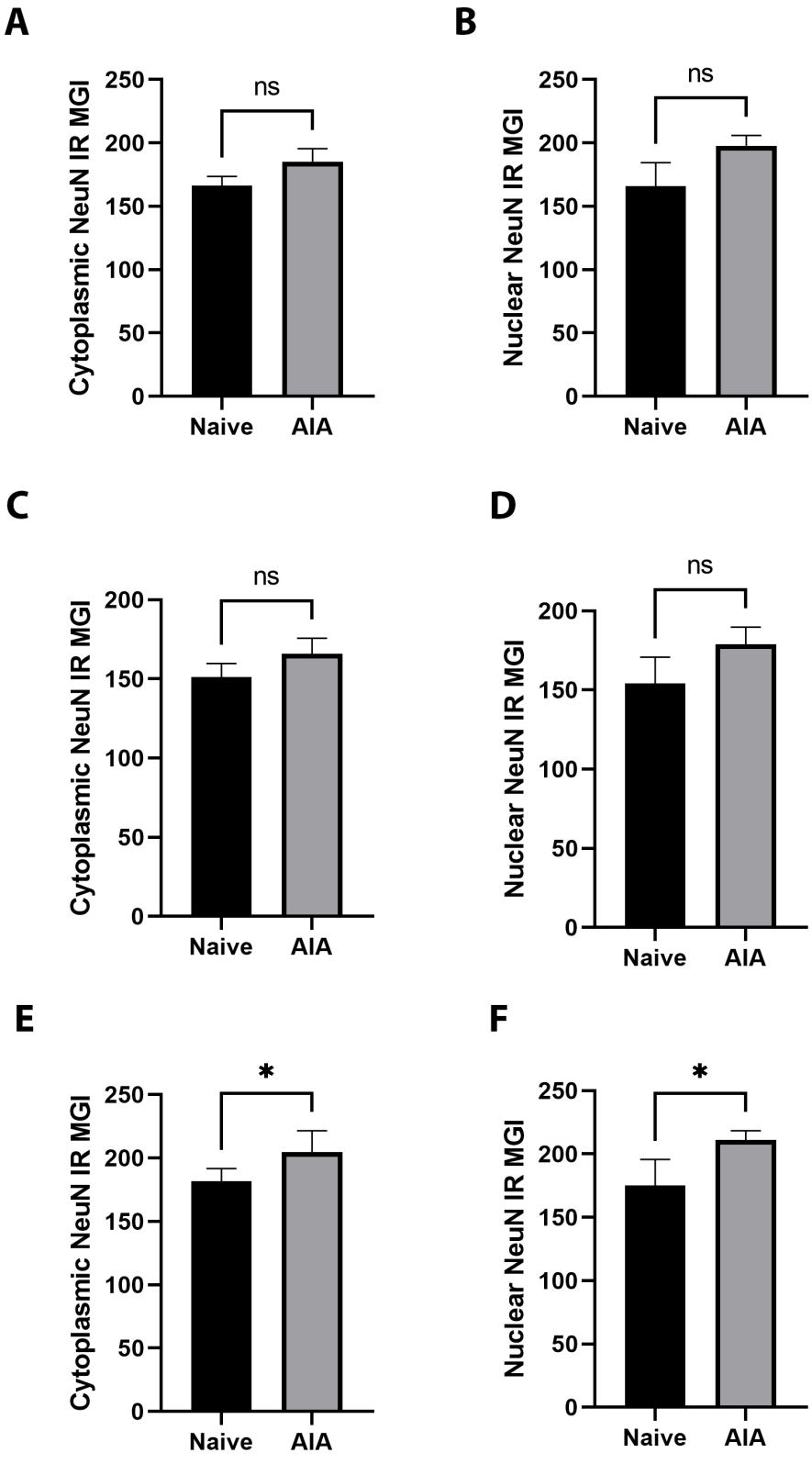

A
B
C
D
E
F
Cytoplasmic NeuN IR MGI
Nuclear NeuN IR MGI
250
200
150
100
50
0
ns
*
Naive
AIA

Figure 2.2 NeuN IR in DRG CGRP subpopulations. Naïve and AIA comparison of NeuN IR in DRG total DRG neurons, CGRP (-) neurons, and CGRP (+) neurons. **A**, no significant differences found between naïve and AIA cytoplasm ($p=0.0571$, $\Delta=17.96$) and **B,** nucleus ($p=0.0571$, $\Delta=39.47$). **C**, no significant differences found between CGRP (-) cytoplasm ($p=0.1143$, $\Delta=14.44$) and **D**, nucleus ($p=0.0571$, $\Delta=27.77$). **E**, significant differences found between CGRP (+) cytoplasm ($p=0.0286$, $\Delta=17.35$) and **F,** nucleus (0.0286, $\Delta=41.15$).

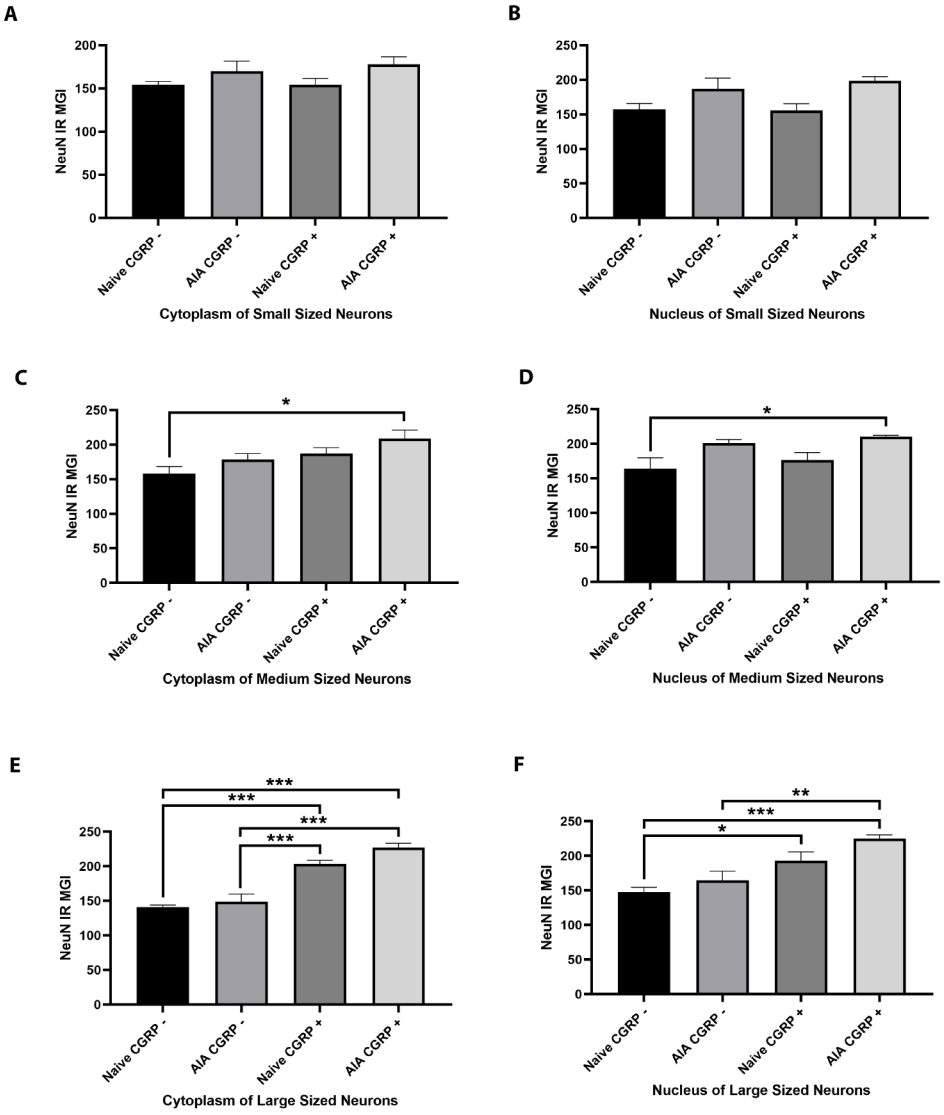


Figure 2.3 NeuN IR between conditions. Evaluation of NeuN IR between condition (naïve/AIA) and CGRP (-) and CGRP (+) neurons. **A-B**, no significant differences found in cytoplasm or nucleus of small neurons. **C-D.** Significant differences were found in the cytoplasm ($F(3, 12)=4.441$, $p=0.026$) and nucleus ($F(3, 12)=4.60$, $p=0.023$) of medium neurons. **E-F**, Significant differences found in cytoplasm ($F(3,12)=36.09$, $p<.001$) and nucleus ($F(3, 12)=11.03$, $p<.001$) of large neurons.

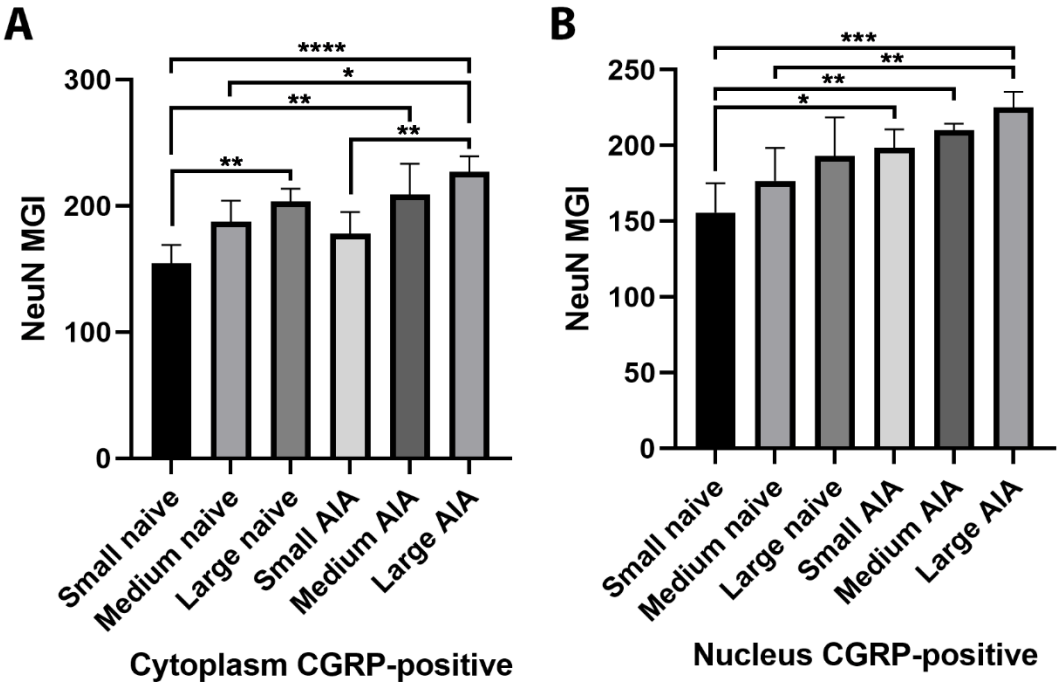


Figure 2.4 NeuN IR between size, condition, and CGRP subpopulations. Significant differences of NeuN-IR between size populations for naïve and AIA groups in CGRP (+) and CGRP (-) DRG neurons. **A**, significant differences found in specific size populations and conditions for CGRP (+) cytoplasm ($F(5, 18)=9.530$, $p=0.0001$). **B**, significant differences found in specific size populations and conditions for CGRP (+) nuclei ($F(5,18)=8.087$, $p=0.0004$).

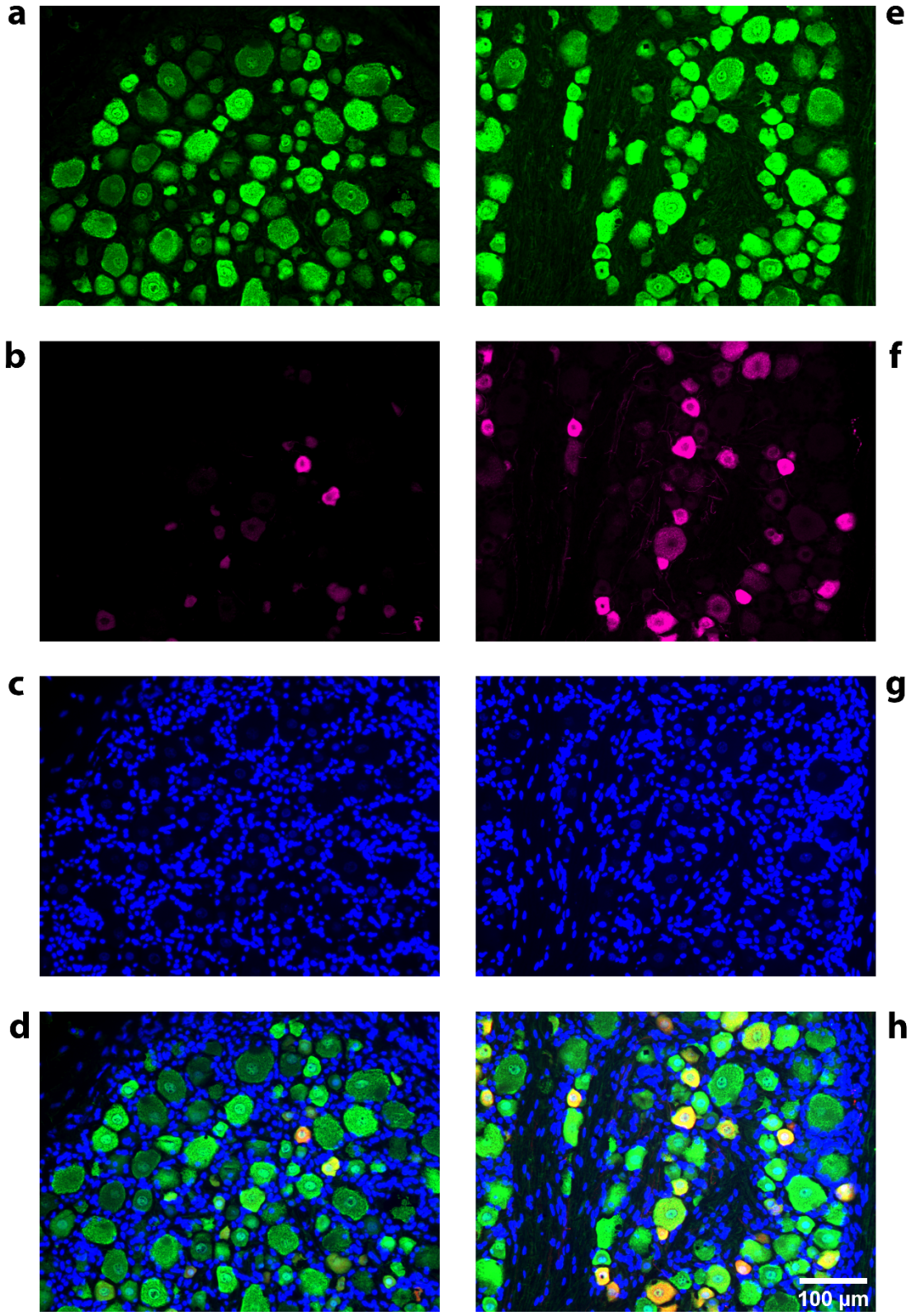

a
e
b
f
c
g
d
h
100 µm

Figure 2.5 Representative fields-of-view for naïve and inflamed AIA conditions. A, NeuN, **B**, CGRP, **C**, DAPI, **D**, naïve composite; and AIA group: **E**, NeuN, **F,** CGRP, **G,** DAPI, and **H,** AIA composite.

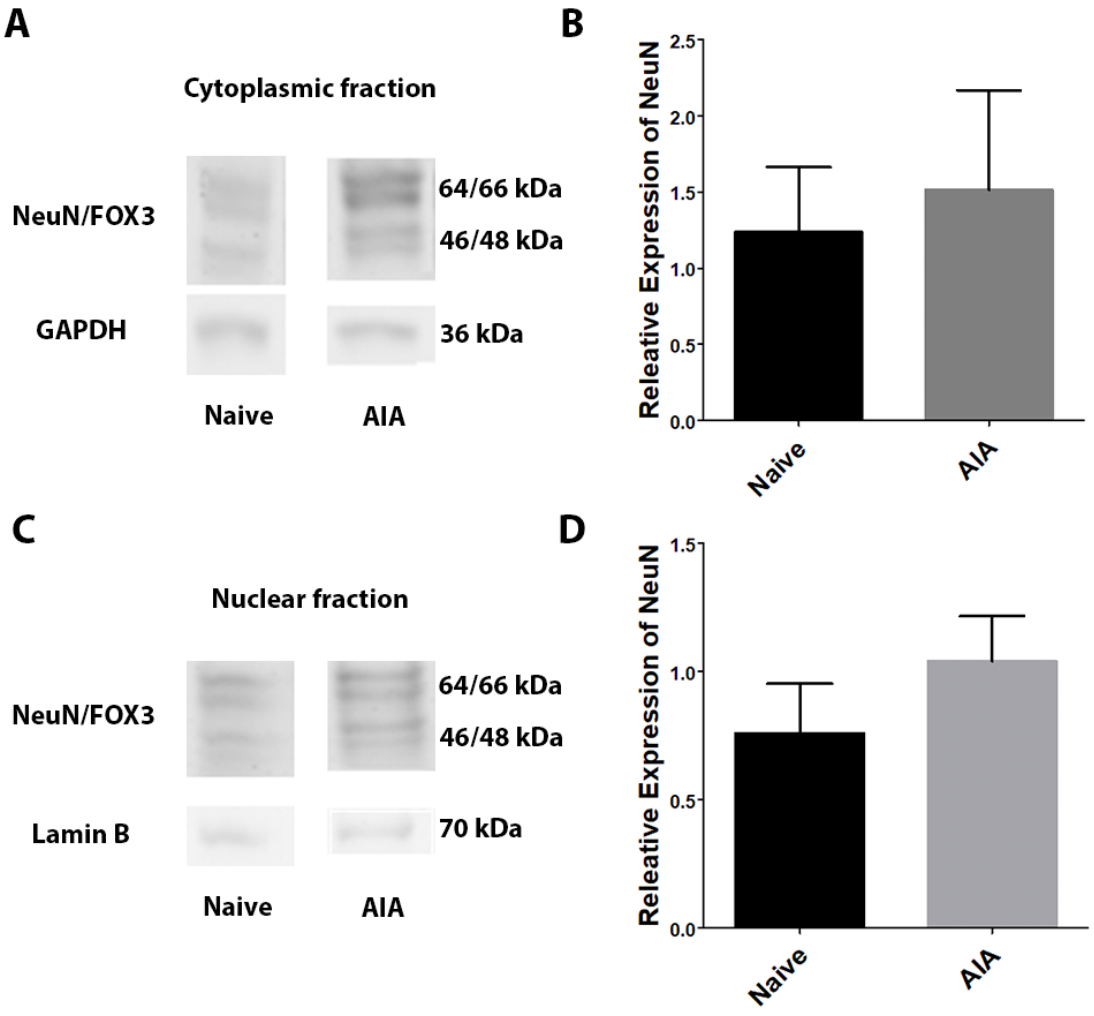


Figure 2.6 Representative naïve and AIA immunoblot of NeuN. A, representative NeuN immunoblot from one naïve rat and **B,** graph of NeuN cytoplasmic fractions. **C**, representative

Location and Size	Test details	Mean 1	Mean 2	Mean Diff.	95.00% CI of diff.	Significant?	Summary	Adjusted P Value	SE of diff.	n1	n2	q	DF
Cytoplasm from **Medium** sized neurons	Naive CGRP - vs. AIA CGRP -	158.2	178.7	-20.51	-62.28 to 21.26	No	ns	0.49	14.07	4	4	2.062	12
	Naive CGRP - vs. Naive CGRP +	158.2	187.3	-29.12	-70.89 to 12.64	No	ns	0.217	14.07	4	4	2.928	12
	Naive CGRP - vs. AIA CGRP +	158.2	208.8	-50.62	-92.39 to -8.850	Yes	*	0.017	14.07	4	4	5.088	12
	AIA CGRP - vs. Naive CGRP +	178.7	187.3	-8.612	-50.38 to 33.16	No	ns	0.926	14.07	4	4	0.8657	12
	AIA CGRP - vs. AIA CGRP +	178.7	208.8	-30.11	-71.88 to 11.66	No	ns	0.196	14.07	4	4	3.026	12
	Naive CGRP + vs. AIA CGRP +	187.3	208.8	-21.49	-63.26 to 20.27	No	ns	0.452	14.07	4	4	2.161	12
Cytoplasm from **Large** sized neurons	Naive CGRP - vs. AIA CGRP -	140.9	148.8	-7.935	-37.26 to 21.39	No	ns	0.852	9.878	4	4	1.136	12
	Naive CGRP - vs. Naive CGRP +	140.9	203.4	-62.56	-91.88 to -33.23	Yes	***	<.001	9.878	4	4	8.956	12
	Naive CGRP - vs. AIA CGRP +	140.9	227.2	-86.34	-115.7 to -57.01	Yes	***	<.001	9.878	4	4	12.36	12
	AIA CGRP - vs. Naive CGRP +	148.8	203.4	-54.62	-83.95 to -25.30	Yes	***	<.001	9.878	4	4	7.82	12
	AIA CGRP - vs. AIA CGRP +	148.8	227.2	-78.4	-107.7 to -49.08	Yes	***	<.001	9.878	4	4	11.22	12
	Naive CGRP + vs. AIA CGRP +	203.4	227.2	-23.78	-53.11 to 5.545	No	ns	0.128	9.878	4	4	3.405	12
Nucleus from **Medium** sized neurons	Naive CGRP - vs. AIA CGRP -	164.1	201.1	-36.99	-78.69 to 4.714	No	ns	0.089	14.05	4	4	3.724	12
	Naive CGRP - vs. Naive CGRP +	164.1	176.6	-12.5	-54.20 to 29.20	No	ns	0.81	14.05	4	4	1.258	12
	Naive CGRP - vs. AIA CGRP +	164.1	210.1	-46.02	-87.72 to -4.315	Yes	*	0.029	14.05	4	4	4.633	12
	AIA CGRP - vs. Naive CGRP +	201.1	176.6	24.49	-17.21 to 66.19	No	ns	0.345	14.05	4	4	2.466	12
	AIA CGRP - vs. AIA CGRP +	201.1	210.1	-9.029	-50.73 to 32.67	No	ns	0.916	14.05	4	4	0.9091	12
	Naive CGRP + vs. AIA CGRP +	176.6	210.1	-33.52	-75.22 to 8.180	No	ns	0.133	14.05	4	4	3.375	12
Nucleus from **Large** sized neurons	Naive CGRP - vs. AIA CGRP -	147.6	164.8	-17.17	-60.08 to 25.75	No	ns	0.646	14.46	4	4	1.68	12
	Naive CGRP - vs. Naive CGRP +	147.6	193	-45.38	-88.30 to -2.467	Yes	*	0.037	14.46	4	4	4.44	12
	Naive CGRP - vs. AIA CGRP +	147.6	225.1	-77.5	-120.4 to -34.58	Yes	***	<.001	14.46	4	4	7.582	12
	AIA CGRP - vs. Naive CGRP +	164.8	193	-28.22	-71.13 to 14.70	No	ns	0.259	14.46	4	4	2.76	12
	AIA CGRP - vs. AIA CGRP +	164.8	225.1	-60.33	-103.2 to -17.41	Yes	**	0.006	14.46	4	4	5.902	12
	Naive CGRP + vs. AIA CGRP +	193	225.1	-32.11	-75.03 to 10.80	No	ns	0.173	14.46	4	4	3.142	12

Table 2.1 NeuN IR between conditions for cytoplasm and nuclei. Post-hoc comparisons using Tukey method for NeuN IR between conditions and CGRP (+) and CGRP (-) DRG neuronal cytoplasm and nucleus, for all size populations. in DRG neuronal sub-compartments and size populations.

Compartment, Condition, and Class	Test details	Mean 1	Mean 2	Mean Diff.	95.00% CI of diff.	Significant?	Summary	Adjusted P Value	SE of diff.	n1	n2	q	DF
Cytoplasm, Naïve, CGRP-positive	Small vs. Medium	154.3	187.3	-33.01	-61.06 to -4.970	Yes	*	0.0232	10.04	4	4	4.648	9
	Small vs. Large	154.3	203.4	-49.12	-77.16 to -21.07	Yes	**	0.0022	10.04	4	4	6.916	9
	Medium vs. Large	187.3	203.4	-16.1	-44.15 to 11.94	No	ns	0.2933	10.04	4	4	2.267	9
Cytoplasm, AIA, CGRP-positive	Small vs. Medium	177.9	208.8	-30.9	-67.83 to 6.025	No	ns	0.1013	13.23	4	4	3.304	9
	Small vs. Large	177.9	227.2	-49.29	-86.21 to -12.36	Yes	*	0.0118	13.23	4	4	5.271	9
	Medium vs. Large	208.8	227.2	-18.39	-55.31 to 18.54	No	ns	0.3855	13.23	4	4	1.966	9
Nucleus, AIA, CGRP-positive	Small vs. Medium	198.4	210.1	-11.76	-30.85 to 7.336	No	ns	0.2505	6.839	4	4	2.431	9
	Small vs. Large	198.4	225.1	-26.72	-45.81 to -7.625	Yes	**	0.009	6.839	4	4	5.525	9
	Medium vs. Large	210.1	225.1	-14.96	-34.06 to 4.134	No	ns	0.127	6.839	4	4	3.094	9

Table 2.2 NeuN IR in size populations between naïve and AIA conditions. Post-hoc comparisons using Tukey method for evaluating size populations within naïve and AIA conditions for CGRP (+) and CGRP (-) DRG neurons.

CHAPTER III

Open-source method of image cytometry in dorsal root ganglion tissue with immunofluorescence

Abstract

Immunohistochemistry (IHC) is a valuable tool in clinical and biological research for evaluating proteins and other antigens in spatially bound tissue. In neuroinflammatory pain research, primary afferent neurons of the dorsal root ganglion (DRG) are studied to understand molecular signaling mechanisms involved in nociception (pain) and inflammation. Measuring IHC (immunofluorescence) in DRG neurons requires manual hand tracing of nuclear and somatic boundaries, which is laborious, error-prone, and may require several weeks to collect the appropriate sample size with a mouse or pen-input display monitor. To overcome these limitations and increase standardization of sampling and measurement, we employed a reliable neuronal cytoplasmic reporter, exclusive to DRG neuronal soma, in a semi-automated algorithm-based approach of Image Cytometry in rat DRG (IC-DRG). The resulting output images are binary nuclear and somatic masks of DRG neurons, defining boundaries of measurement for CellProfiler and manually

scored at 94% accurate. Herein, we successfully show a novel approach of automated image analysis for DRG neurons using a robust ImageJ/FIJI script, overcoming morphological variability and imaging artifacts native to imaging frozen tissue sections processed with immunofluorescence.

Introduction

Immunofluorescence is a powerful analytical tool for measuring subcellular proteins in two-dimensional and volumetric tissue imaging protocols. IHC image analysis traditionally requires hand tracing of micrographs, manually defining cytoplasmic and nuclear boundaries for measurement using a mouse or pen-input display monitor (Miller et al., 1993; Suzuki et al., 2010; Zhang, 2013; Hoffman et al., 2016). This process is effective although may take several days to weeks of manual tracing and is susceptible to undetected human error. There are a variety of commercial methods to automate the measurement of cells in images captured from cell culture plates (Carpenter et al., 2006; Kamentsky et al., 2011; Yufera et al., 2011; Georg et al., 2018; Czech et al., 2019); yet, automated image analysis in tissue has been elusive, due to heterogenous cell populations, proximity of cells, morphological variability, available biomarkers, and nonspecific fluorescent artifacts.

Divergent tissue types present unique obstacles for developing accurate methods of algorithm-based cell-specific detection. Pan-nuclear biomarkers, such as 4′,6-diamidino-2-phenylindole (DAPI), are often employed in cell culture techniques to label and automatically detect a single cell type, however, the same biomarker in spatial, heterogenous tissue additionally labels the nuclei of cells outside of the study. For example, in DRG tissue, DAPI labeled neurons (cells of interest) have a medium to large cross-sectional area and broad range of intensity, while satellite support cells (cells of non-interest in this study) have a small cross-sectional area, tightly surround/outnumber every neuron, and all elicit intense DAPI emission. I hypothesize that a

second biomarker may be employed to overcome histological challenges by labeling the cytoplasm of a cell type of interest, thereby allowing for algorithmic detection of cytoplasm and nuclei from a specific cell type, automatically. In this study, neuronal nuclei (NeuN) antigen was selected for consistent and reliable DRG neuronal cytoplasmic, non-axoplasmic, antigenicity (Anderson et al., 2020). The IC-DRG script was developed in ImageJ 1.x to standardize camera exposure and output binary nuclear and somatic masks of DRG neurons, based on DAPI and NeuN immunoreactivity (IR).

Methods

Animal Model

Male and female Sprague Dawley rats (Charles River; n=9, 150-250 grams, 8 weeks of age) were bred, maintained on-site, and used to evaluate the IC-DRG script with images collected from in-vivo IHC immunolabeling, in naïve and inflammatory conditions. The main study consisted of four male and four female rats (n=8) and a smaller study to measure the accuracy of IC-DRG processing with a traditional fixative consisted of one rat (n=1). All rats were given food and water *ad libitum* and housed with a 12-hour on/off light cycle. All methods and techniques utilized in these experiments were conducted in accordance with the National Institute of Health (NIH; https://www.ncbi.nlm.nih.gov/books/NBK43327/), the International Association for the Study of Pain (IASP; https://www.ncbi.nlm.nih.gov/pubmed/6877845) and approved by the Oklahoma State University Center for Health Sciences Institutional Animal Care and Use Committee (2016-03). All efforts were made to care for and minimize the total number of rats used in these experiments to reduce potential suffering.

Model of Inflammation

The cytoplasmic expression pattern of NeuN IR is important for accurate algorithmic processing, accordingly, the potential range of NeuN IR expression in DRG neuronal cytoplasm was evaluated during naïve conditions and at a peak swelling timepoint of inflammation, 48-hours, in the adjuvant induced arthritis model (AIA) (Whiteley and Dalrymple, 2001; Miller et al., 2012; Zhang, 2013; Hoffman et al., 2016). Complete Freund's adjuvant (CFA) (Sigma, F5881) was emulsified in a 1:1 ratio with phosphate buffered saline (PBS) and injected into the right hind paw of experimental animals to establish unilateral adjuvant-induced arthritis. All rats (n=9) were housed for 48-hours with food and water provided *ad libitum*. The naïve group included two male and two female untreated rats. The AIA group included two male and two female rats anesthetized with isoflurane (1.5 liters/minute oxygen, 2% isoflurane) and injected with 150 µl emulsified CFA in the center of the right hind paw with a 26-gauge needle. The metatarsal region of the right hind paw of all animals was measured with a dial caliper (Mitutoyo; Aurora, IL, USA) shortly before transcardial perfusion (Hoffman et al., 2010).

Immunofluorescence

After 48 hours, naive/AIA rats (n=4, n=4) were deeply anesthetized with avertin, 1 gram (2, 2, 2, tribromoethanol; Sigma, T48402) dissolved in 1 ml 2-methyl-2-butanol (Sigma, 240486), diluted to 2.5% (*v/v*) in PBS (pH: 7.3) and administered intraperitoneally (IP). To determine level of anesthesia, a three-point flinch test was performed on the eye, tail, and hind paw. Once rats were determined unresponsive, 1.0 ml xylazine (Anased, 100mg/ml) was delivered IP and approximately 1 minute later rats were transcardially perfused with 100 ml calcium-free tyrode solution (pH: 7.3) to flush the vascular system, followed by transcardial perfusion with fixative. Transcardial perfusions were performed with a peristaltic pump (Cole Parmer, 7567-70) at a speed of four (37 ml/minute) and a total volume of 425 ml of fixative, per rat, to provide optimal

histologic morphology. Rats were perfused with a modified Zamboni's fixative: 0.75% (*w/v*) picric acid, 0.2% (*w/v*) paraformaldehyde (PFA), phosphate buffered saline (PBS, pH: 7.3) (Hoffman et al., 2010). This low aldehyde fixative has been shown to provide optimal labelling in DRG neurons (Hoffman et al., 2010). The right ipsilateral lumbar 4 DRG were collected and post-fixed for 3 hours at 4°C and transferred into PBS with 10% (*w/v,* pH 7.3) sucrose, overnight (Lin et al., 2011; da Silva Serra et al., 2016; Hoffman et al., 2016). Sucrose cryopreservation assists in preserving the integrity of cellular morphology during freezing. A common technique for immunofluorescence is perfusion with 4% PFA (highly crosslinks proteins), followed by incubation in 15% sucrose, and 30% sucrose (to reduce osmotic morphological shrinkage). In our study, rats were perfused with of a modified Zamboni's fixative (0.75% (*w/v*) picric acid, 0.2% (*w/v*) PFA) and followed by incubation in 10% sucrose (overnight at 4°C). Based on previous research, a lower concentration of sucrose (10%) appeared to be optimal in tissue fixed with a lower concentration of PFA (0.2%) (Hoffman et al., 2010; Hoffman et al., 2011).

All tissues were placed into a single mold with M1 embedding matrix (Thermo Fisher Scientific, 1310TS) and frozen with liquid nitrogen. Frozen DRG sections (12 µm) were cut on a Leica cryostat (Leica, CM 1850-3-1) and thaw-mounted on gelatin-coated glass slides. All slides were dried for 60 minutes on a slide warmer set to 37°C before antibody incubations and PBS washing steps. Primary antiserum was diluted in PBS with 0.5% (*w/v*) bovine serum albumin (BSA) (Sigma Cat# A7906-100G), 0.5% (*w/v*), polyvinylpyrollidone (PVP) (Sigma Cat# P5288-100G), and 0.3% (*v/v*) Triton-X 100 (Sigma Cat# X100-500ML), pH: 7.3. Chicken polyclonal anti-NeuN (Millipore Cat# ABN91, RRID: AB_11205760) was diluted 1:1000. Mouse monoclonal anti-CGRP was diluted 1:2000 (Santa Cruz Cat#: 57053, RRID: AB_2259462). Rabbit polyclonal anti-glutaminase (GLS, Millipore Cat# ABN1466, RRID: AB_2861206) was diluted 1:2000. Nuclear staining was achieved with DAPI at a concentration of 300 nM (Thermo Fisher Scientific Cat# D3571, RRID: AB_2307445). Primary antibodies were combined for multiplex labeling in

PBS-BSA-PVP-Triton-X (pH, 7.3). All primary antibodies for IHC are verified, by their manufacturer, to be specific and reactive to rat. Slide containers were sealed with parafilm to prevent evaporation and placed on a rocker at 4°C for 96 hours. Slides were washed three times in PBS for 10 minutes each at room temperature to remove unbound primary antiserum. Tissue was incubated in donkey anti-chicken 488 (Jackson ImmunoResearch Labs Cat# 703-545-155, RRID: AB_2340375), donkey anti-mouse Alexa Fluor 647 (Thermo Fisher Scientific Cat# A-31571, RRID: AB_162542), and donkey anti-rabbit 555 (Thermo Fisher Scientific Cat # 31572; RRID: AB_162543). Secondary antibodies were diluted to 1 µg/ml (1:1000) in PBS with 0.3% (*v/v*) Triton-X, combined for multiplex labeling, and incubated on a rocker for one hour at room temperature. During secondary and DAPI incubation, and subsequent incubation steps, all slide containers were covered with aluminum foil and placed on a rocker to prevent fluorophore bleaching from ambient light. Afterward, slides were washed three times in PBS, incubated in 300 nM DAPI in PBS for 15 minutes and washed three more times in PBS before cover-slipping in Prolong Gold anti-fade mounting medium (Thermo Fisher Scientific Cat# P36930).

To determine whether images collected from a traditional fixative affect IC-DRG processing, tissue collected from one naive rat perfused with 4% (*w/v*) PFA in PBS, pH: 7.3, was evaluated. A single naïve rat (n=1) was perfused with 4% PFA for manual scoring with a traditional fixative, in addition to 4 naïve and 4 AIA (n=8) rats perfused with a modified Zamboni's fixative in the main study (camera exposure optimization, IHC comparisons between manual hand tracing and IC-DRG processing, manual scoring, and size distribution). Collected DRGs, perfused with 4% PFA (n=1), were sectioned and immuno-labeled with chicken α-NeuN (Millipore Cat# ABN91, RRID: AB_11205760), mouse α-CGRP (Santa Cruz Cat#: 57053, RRID: AB_2259462), and rabbit polyclonal anti-glutaminase (GLS, Millipore Cat# ABN1466, RRID: AB_2861206), labeled with DAPI (Thermo Fisher Scientific Cat# D3571, RRID: AB_2307445). Previously

described IHC processing steps for modified Zamboni's fixative were also followed for processing tissue perfused with 4% PFA.

Imaging

All images were collected on a BX51 epifluorescence microscope (Olympus; BX51TRF) with uplanFL 20x/0.50 ∞ 0.17 objective, using a SPOT RT470 camera (Spot Imaging, 7.4 Slider, pixel size=7.4 µm x 7.4 µm) at a pixel resolution of 1600x1200 and saved in *.tiff format. The collected and uncompressed 8-bit images have a mean grey intensity (MGI) range of 0-255, for 256 incremental values. Camera exposure was calibrated for each filter channel, so the total average of naive neurons had an MGI equal to 125±5. One image per channel was taken for NeuN (exposure: 250 milli-seconds (ms); gain: 2), DAPI (exposure: 400 ms; gain: 2), CGRP (exposure: 2.5 seconds; gain: 2), and GLS (exposure 1 ms; gain: 2) channels. The sample size evaluation contained 29 images per rat (modified Zamboni fixation, naïve=4, AIA=4) and other studies (camera exposure optimization, immunofluorescence (IR) comparisons between manual hand tracing and IC-DRG processing, manual scoring, and size distribution) contained 16 images per group (modified Zamboni fixation, naïve=4, AIA=4). The sample size evaluation contained 29 images per rat and the standard error of the mean (SEM) was measured sequentially, from 2 to 29 images, to determine the optimal number of images for an appropriate sample size, based on cytoplasmic NeuN IR. To evaluate a traditional fixative, one additional naïve rat (traditional fixative, n=1) was perfused with 4% PFA, 16 images were captured, and manually scored for error and compared with the manually scored error for tissue processed by the Zamboni's fixative.

Manual Tracing

For each field of view, every DRG neuronal cell body (soma) that was fully within the given field of view was carefully hand traced in Image J software version 1.48v (Schneider et al., 2012)

using a Wacom Cintiq 21UX with a pressure sensitive pen display (https://www.wacom.com). Manually defined regions of interest were measured for size and NeuN IR expression in DRG neuronal cytoplasm. Manually defined regions of interest for each neuron was defined by manual hand tracing of DRG neuronal nuclei and cytoplasm.

IC-DRG Script

The IC-DRG script was written in the ImageJ Macro language on the Windows operating system and is composed of over 2,000 lines of ImageJ/FIJI filtering functionality. Set-up of the IC-DRG package required the free and NIH funded, software packages: ImageJ/FIJI (Schneider et al., 2012), Adjustable Watershed plugin (Watershed, 2019), and CellProfiler (Carpenter et al., 2006; Kamentsky et al., 2011). The IC-DRG script is run in the ImageJ scripting window and initialized a graphical user interface window to assign the path of the IC-DRG processing subfolders. First time assignment of folder paths required a somewhat lengthy (~< 2 minutes) initial folder pathway assignment process, but subsequent use did not, if the names of the folders or their location were unchanged.

The IC-DRG script defines multiple size threshold values, according to a 'set scale' function (micron/pixel) embedded within each macro of the script. The Spot camera in this study captures images with a pixel size of 7.4 µm x 7.4 µm and a set scale value of 2.702 micron/pixel, for a 20x objective. The set scale value for images captured with other objectives could be calculated for the correct pixel size scale (µm x µm) and updating the script; however, preliminary research indicated that the 20x objective was optimal for DRG neurons. Images captured at 10x resulted in more counted objects than 20x, increased clumping (error), and low resolution. Images captured at 40x resulted in fewer neurons per image (than 20x), increased intracellular IR variability, and ultimately the need to capture many more images to attain the appropriate sample size (due to a lack of neurons per field of view). However, the IC-DRG script could be modified for images

captured with 10x, 40x, 60x, or 100x objectives, for other studies. The official ImageJ forum (https://forum.image.sc/) is a valuable resource to learn and ask questions about ImageJ scripting, ImageJ plugins, and CellProfiler.

The IC-DRG script generated binary output masks of DRG neuronal nuclei and somata, based on NeuN IR and DAPI emission. The IC-DRG script requires two filter channels, one dedicated to NeuN and one dedicated to DAPI fluorescence, additional channels may be included to evaluate fluorescence markers of primary interest. Binary output masks, and other channels of interest (CGRP and GLS), were imported into CellProfiler for measurement. GLS IR (exclusive to DRG neuronal cytoplasm) was included in this study for merging with IC-DRG-generated nuclear and cytoplasmic demarcations (via CellProfiler), optimizing for contrast to illustrate manually scored error for publication. GLS is not required, or suggested, to assist in manually scoring of images. Binary masks generated from the IC-DRG script functioned as regions of interest for defining nuclear, cytoplasmic, and somatic boundaries of DRG neuronal measurement of DAPI, NeuN, CGRP and GLS channels. Additional filter channels may be processed in CellProfiler, with the IC-DRG masks, depending on the number of available fluorescence filters available for imaging.

Manual Scoring

Images collected from the naïve group were manually scored to determine percent error and percent accuracy. For every field of view evaluated in CellProfiler, exported duplicate images from the GLS channel were exported with cellular boundaries overlaid and manually scored to determine total count and total error. GLS IR is largely absent in DRG neuronal nuclei and was included in this study to increase nuclear and cytoplasmic contrast for publication; however, the evaluation of NeuN alone provides quick and easy manual scoring of error. GLS IR was included to enhance cytoplasmic and nuclear contrast for illustrating manual scored error and to exhibit IC-DRG processing with four channels. Following manual scoring, each erroneous object was

measured and categorized as small, medium, large, CGRP-positive, or CGRP-negative. Individual erroneous objects were cropped and their automatically assigned subcellular DRG neuronal demarcations were qualitatively evaluated for type 1 error.

Camera Exposure Optimization

The IC-DRG script contains a predefined set of functions for processing, based on cytoplasmic NeuN IR pixel intensity, in rat DRG neurons imaged at 20x with epifluorescence microscopy. The IC-DRG script contains algorithms compatible with a wide range of NeuN IR pixel intensity variation, however, developing a strategy to calibrate the camera exposure for the optimal range was important, and due to this, determining a range of optimal image brightness was important. The IC-DRG script is compatible with a wide range of pixel intensity variation but a strategy to identify and report the range was also important. To achieve this, an evaluation mean grey intensity (EMGI) script was created and embedded into IC-DRG functionality (Anderson, 2018). The EMGI code was developed to measure overall mean pixel-intensity of each image and calculate an evaluation mean grey intensity (EMGI) score for DAPI and NeuN channels. The EMGI score is a gross measurement of camera exposure and used to optimize camera exposure for IC-DRG processing. The EMGI script measures the total pixel intensity for the entire field of view, in a range between 30 – 255 histogram units (8-bit), allowing for camera exposure calibration of NeuN and DAPI channels with a small number of images for evaluation, before committing to photographing the entire experiment. The EMGI score was determined by 6 DAPI and 6 NeuN images per condition, containing mostly DRG tissue with little to no empty (black) space. I suggest capturing 6 images per condition, with mostly DRG tissue (can include ganglion boarders but otherwise little to no empty space), to attain a robust sample size of neurons for EMGI calibration. Images collected for the main study (outside of calculating the EMGI score) may contain varying amounts of tissue per image, as images with less DRG tissue this will simply result in fewer neurons per image. The IC-DRG script generates a spreadsheet *.csv file from

which the EMGI score is easily calculated to determine the estimated optimal camera exposures for NeuN and DAPI channels.

Measurement of the same field of view with 13 different camera exposures would result in deleterious photobleaching. To overcome this limitation, the pixel intensity variation (PIV) script was developed to increase or decrease the brightness and contrast of every calibrated (125±5 MGI units) field of view, across 12 variations of pixel intensity, dim to bright. The PIV script automatically adjusts the original pixel intensity, in increments of 20 histogram levels (8-bit; 0-255 levels of intensity), across 12 intensity variations, such as: -120, -100, -80, -60, -40, -20, 0 (original image intensity), +20, +40, +60, +80, +100 and +120 (Anderson, 2019). Each batch of images were processed by IC-DRG script to generate an EMGI score per image, record total count per image, and manually score each image for type 1, false positive, error identification. Camera exposure optimization charts for DAPI and NeuN were plotted from PIV script data to illustrate how EMGI scores affect algorithmic accuracy and total count. Referencing EMGI scores in the optimization charts allow users to calibrate the microscope camera exposure for NeuN and DAPI channels, for optimal algorithmic image processing.

Sample Size Evaluation

The total number of images per naïve rat DRG (n=4) was 29 (116 total images). Images were taken from 5 slides with 3 DRG sections per slide. All images for the sample size evaluation were sequentially measured in ascending order of image batches, 2 to 29, and the SEM was collected from each batch to identify an optimal number of images, per rat, to attain a proper sample size. The grand mean of NeuN IR in DRG neuronal cytoplasm were collected from each image batch processed by IC-DRG and organized into separate *.csv files for processing, such as, the first *.csv file contained the grand mean data from 2 images, the second *.csv file contained the grand mean data from 3 images, and so forth, up to the 28th *.csv file, containing the grand mean for all

29 images. A script was developed in R to process each *.csv file and generate confidence intervals for plotting and evaluation towards determining the number of images required to provide a robust sample size. NeuN IR in DRG neuronal cytoplasm was randomly sampled 1,000 times, with replacement, using the Boot library package (https://cran.r-project.org/web/packages/boot/index.html) to calculate the SEM and 95% confidence intervals. The SEM was sequentially plotted for all 28 *.csv files, representing data from 2 through 29 images per-channel. Bias-corrected and accelerated (BCa) bootstrap confidence intervals were plotted with a linear trend line from data collected in 20 through 29 images per-channel, comparing data collected from at-least 20 images per-channel to data collected from 29 image per channel. SEM and BCa confidence intervals calculated from NeuN IR measured in DRG neuronal cytoplasm was evaluated to determine an estimate for the number of images that may provide a robust sample size per group.

Size Distribution

DRG neuronal size populations were categorized in a trimodal size distribution of small (cross-sectional area 0-400 μm^2; diameter 0-23 μm), medium (cross-sectional area 400-800 μm^2; diameter 23-32 μm) and large (cross-sectional area 800+ μm^2; diameter 32+ μm) (Kawamura and Dyck, 1978; Lawson and Biscoe, 1979; Lawson et al., 1984). The cross-sectional area of all groups was plotted in a size distribution histogram with a smooth density estimate curve. Previous reports have defined a characteristic size distribution of DRG neurons, featuring a strong peak from 400 – 500 μm^2 and another smaller prominence around 1,400 μm^2 (Kawamura and Dyck, 1978; Lawson and Biscoe, 1979; Lawson et al., 1984).

Statistical Analysis

Power analysis (α=0.05, power=0.80, mean 1=100, mean 2=120) indicated a sample size of n=4 for each of our groups (https://clincalc.com/stats/samplesize.aspx), therefore n=4 for naïve and

n=4 for AIA. Statistical analysis were processed in R (R Core, 2013) and Graphpad Prism 8 (www.graphpad.com). Hind paw edemas were statistically evaluated in R by comparing the metatarsal means from naïve (n=4) and AIA (n=4) conditions. The grand mean of NeuN IR for each rat was analyzed with Prism 8 for normal distribution by the Shapirio-Wilk normality test and statistically evaluated by one-way ANOVA with Tukey post-hoc multiple comparisons. Significance was determined when the p-value was less than 0.05.

The grand mean of NeuN IR in naïve DRG neuronal cytoplasm were tested in R for normal distribution by Shapiro-Wilk testing and evaluated for correlation by calculating the Pearson coefficient, between techniques. Significance on non-correlation was determined when the p-value was less than 0.05.

Standard deviation (SD), SEM, and 95% BCa confidence intervals of DRG neuronal cytoplasm were calculated and plotted in R. SD was reported in the hundredths for results between 0.1 and 1 and in the tenths for results between 1 and 100. Manually scored percent error was calculated by dividing total identified error by the total neuronal objects identified and multiplying by 100.

$$\frac{Total\ Count\ of\ Errors}{Total\ Count}\ X\ 100 = Manually\ Scored\ Percent\ Error$$

Percent accuracy was calculated by subtracting 100 from manually scored percent error.

$$\frac{(Total\ Count) - (Total\ Count\ of\ Error)}{Total\ Count}\ X\ 100 = Manually\ Scored\ Percent\ Accuracy$$

Pooled SD was calculated by squaring the residual mean square in Graphpad Prism. The ImageJ 'set scale' function value for the 20x objective was calculated from the pixel size (μm^2) resolution of images (pixel size value for camera should be on manufacture's website).

$$\frac{Magnification\ of\ Objective}{Pixel\ Size\ (\mu m\ x\ \mu m)} = \ value\ to\ set\ scale\ in\ IC - DRG\ script$$

Code & Data Accessibility

All figures and tables in this study may be replicated with the original images, text/video tutorials, scripts, and data files (https://github.com/MichaelBAnderson) . Current IC-DRG (v1.41) and PIV script packages are compatible with the latest (to date) release of ImageJ/FIJI (v1.53f) and is released under GNU General Public License v3. The IC-DRG package, all future updates may be found at https://github.com/MichaelBAnderson/IC-DRGs_v1.41_FIJI_script. A *required* "adjustable watershed" plugin (https://imagejdocu.tudor.lu/plugin/segmentation/adjustable_watershed/start) is provided in the IC-DRG_v1.41_FIGI_script folder hierarchy (Watershed, 2019). The PIV script may be found at https://github.com/MichaelBAnderson/Image_Intensity_Variation_script. All data generated for this manuscript may be found at https://github.com/MichaelBAnderson/IC-DRGs_Data. Two video tutorials for user operation of the IC-DRG script and data organization may be found at https://github.com/MichaelBAnderson/IC-DRGs_Tutorials. All images may be found at https://github.com/MichaelBAnderson/IC-DRGs_Imagesets. All files were compressed by WinRAR archiver (https://www.rarlab.com).

Results

Model of Inflammation

There was a significant difference (p=0.0286) in metatarsal thickness between naïve (n=4, mean=3.45 mm, SD=0.55) and AIA (n=4, mean=8.11 mm, SD=0.28). The hind paw was increased by 2.35-fold, from naïve to AIA rats.

Imaging

All rat L4 DRG neuronal cell bodies were immunoreactive for NeuN antibodies. The naïve group exhibited more of a range in somatic, non-axoplasmic, NeuN expression between neurons than

the AIA group, which were intensely immunoreactive for NeuN (**Figure 3.1**). DRG neuronal nuclear and cytoplasmic boundaries were defined for manual hand tracing with DAPI and DRG neuronal cytoplasmic glutaminase immunoreactivity, respectively (**Figure 3.2**).

IC-DRG Script

The IC-DRG (v1.41) script, developed in ImageJ 1.x scripting language, automatically produced accurate binary mask images of DRG neuronal nuclei and soma with minimal error (**Figure 3.9, A; Figure 3.11, A**). Accuracy was achieved through strict size, shape and MGI intensity requirements to maintain quality of data (**Figure 3.3**). The resulting masks were imported into CellProfiler to measure DAPI, NeuN, CGRP, and GLS channels, in nuclear, cytoplasmic, and whole DRG neuronal compartments. (Carpenter et al., 2006; Kamentsky et al., 2011). IC-DRG processing was designed for images captured at 20x and configured to exclude objects touching any corner of the field of view, discarding incomplete neurons. IC-DRG processing is based on an adjustable 'set scale' function and may be modified for other objectives; however, it was determined that 20x is optimal for count and subcellular resolution of pixel intensity in DRG neurons. Data for calculating EMGI scores are exported into a *.csv file for each field of view (**Figure 3.4, F**).

The *required* "adjustable watershed" plugin and a complete list of instructions, video tutorials, and online links for running IC-DRG as an ImageJ 1.x script (v.1.52n) are located in the documents folder (\\IC-DRGs\Documents) of the IC-DRG package (Anderson, 2018) (**Figure 3.4, B**), and Github (https://github.com/MichaelBAnderson/IC-DRGs_Tutorials). CellProfiler Analyst is additional software that further offers machine-learning capabilities for algorithmic training and accuracy optimization; however, this software requires special installation, therefore, in this study, I will only discuss CellProfiler.

Manual Scoring

Each manually scored image consisted of a GLS channel and the corresponding IC-DRG mask-derived demarcations (**Figure 3.5**). The GLS channel, primarily neuronal cytoplasmic IR, was merged, instead of another channel (such as NeuN), with IC-DRG-derived demarcations to increase visual contrast of subcellular compartments for publication. NeuN IR is found throughout DRG neuronal soma, and the resulting pixel intensity (white color) increases the green (nuclear) and red (cytoplasmic) IC-DRG-derived boundaries, providing contrast for manual scoring error; however, any relevant channel may be selected for manually (visually) scoring error. The quality of data resided in three categories: positive, negative, and mixed. Positive category neuronal objects included optimal automated nuclear and somatic demarcations (**Figure 3.5, A-C**). Mixed category neuronal objects included automated demarcations with sub-optimal accuracy; nevertheless, many neuronal objects of the mixed category remained reflective of the type of neuron the demarcations were being assigned (**Figure 3.5, D-F**). Negative category neuronal objects included skewed boundaries and were not reflective of actual neurons (**Figure 3.5, G-I**). In this study, conservative scoring of the mixed category reflected strict scoring requirements.

Percent accuracy for the total naïve neuronal population (n=4, modified Zamboni's fixative) was manually scored at 93.81% (SD=1.78). The total neuronal count identified by IC-DRG processing was 648, compared to 710 for total manually identified neurons. Small and medium size populations were manually scored at 96% and 95% percent accuracy, respectfully, large size neurons were scored at 90% accurate (Table 2). While IC-DRG is more accurate for small and medium neurons, qualitative evaluation of every false positive error appears to illustrate random sources of errors from every population (**Figure 3.13**). Qualitative evaluation of IC-DRG processing from 16 control images (n=1) captured from a naïve rat transcardially perfused with 4% PFA (traditional) fixative, resulted in 93.68 % accuracy.

Immunofluorescence

There were no significant differences of NeuN IR in DRG neuronal cytoplasm for the naïve group, between technique or size population, except for naïve CGRP (+) ($F(5, 17)=7.903$, $p=0.0005$, pooled SD=14.7) and AIA CGRP (+) ($F(5, 18)=11.29$, $p<0.0001$, pooled SD=15.4) (Figure 7). There were no significant differences between the same size population (small vs small, medium vs medium, large vs large) with Tukey post hoc multiple comparison analysis (**Figure 3.6, Table 3.1**).

There were significant correlations of NeuN IR in DRG neuronal cytoplasm, determined by Pearson correlation coefficient, between manual hand tracing and IC-DRG processing techniques for naïve ($r = 0.8301$; $p= 0.1699$) and inflammatory ($r = 0.9293$; $p= 0.0707$) conditions.

There were no observable differences in neuronal shrinkage or manually scored error between fixatives (modified Zamboni's vs. 4% PFA) or a nontraditional sucrose concentration (10% sucrose), indicating that traditional IHC protocols (4% PFA, 30% sucrose) should be compatible with IC-DRG processing. The key to optimal image processing with the IC-DRG algorithm is preserved morphology and contrast of the DAPI and NeuN fluorescence signal, with low background. Different processing methods to achieve high quality images should be compatible with IC-DRG processing.

Camera Exposure Optimization

The original batch of NeuN IR and DAPI images were processed by the pixel intensity variationPIV script, generating 12 new batches of images (-120, -100, -80, -60, -40, -20, 0 (original image intensity), +20, +40, +60, +80, +100 and +120) for each channel. The total number of image batches for NeuN IR (12 plus original) and DAPI (12 plus original) were processed individually by the IC-DRG script to generate EMGI values for each batch, determine total count for each batch, and total error for each batch of images. Data gathered from all

variations of NeuN IR (**Figure 3.8**) were evaluated for manually scored error (**Figure 3.9, A**) and total count (**Figure 3.9, B**), and plotted to determine camera exposure optimization parameters. Data gathered from all variations of DAPI MGI (**Figure 3.10**) were evaluated for manually scored error (**Figure 3.11, A**) and total count (**Figure 3.11, B**), and plotted to determine camera exposure optimization parameters. The optimal range of camera exposure was determined for NeuN IR between 34 – 62 EMGI and for DAPI between 72 – 103 EMGI (**Figures 3.9, 3.11**).

Sample Size Evaluation

The average SD calculated from 20 images per-channel was 3.87 and the average SD calculated from 29 images per-channel was 2.92, a difference of less than 1 SD value. SEM for all groups were greatly reduced when processing 20 through 29 images per-channel (**Figure 3.12; *1-*1**). 95% BCa confidence interval plots indicate a tight SD of resampled means in data collected from 20 through 29 images per-channel (**Figure 3.12; *2-*2**). Collectively, these results indicate that 20 images per channel (n=4) provides a robust sample size for IC-DRG processing.

Size Distribution

Consistent with previous reports, a prominent peak of DRG neurons were detected for both techniques around 400-500 μm^2, and smaller prominence around 1,400 μm^2 (Kawamura and Dyck, 1978; Lawson and Biscoe, 1979; Lawson et al., 1984) (**Figure 3.13**). These findings suggest that IC-DRG processing yields a previously reported sampling size distribution, and consistent with the manual hand tracing.

Discussion

Conventional quantitative methods of measuring immunofluorescence in DRG neurons are time consuming, open to user bias, and may be prone to undetectable error. On the other hand, an accurate and reliable method of automated image analysis for IHC fluorescence has been elusive

for many reasons. The DRG, as an example, is neuronal tissue containing a heterogeneous cell population of primary sensory neurons, satellite glial cells, and immune cells. The fluorescence emission of DAPI labeling in DRG tissue is characterized by low to intensely labeled DRG neuronal nuclei, tightly surrounded by intensely labeled satellite cell nuclei. A heterogeneous cell population, contrasting emission of DAPI in (and proximity of) satellite cells and neurons, the range of NeuN IR within individual DRG neurons, and diverse morphology all posed challenges to developing an accurate algorithm for image cytometry in DRG neurons. Overcoming these challenges required algorithmic image processing based on NeuN IR and DAPI fluorescence to generate binary masks of DRG neuronal nuclei and somata, which were measured in CellProfiler with other proteins of interest.

The IC-DRG script processes images based on fluorescence pixel intensity. Therefore, a method of standardizing camera exposure between experiments, users, and collaborating labs was required. The optimal ranges of camera exposure were reported in EMGI-based camera exposure optimization charts (Figures 8, 10). Manual scoring of images exported from CellProfiler confirmed that IC-DRG processing was accurate over a broad range EMGI-scores.

Regardless of technique, preserved cell morphology is required for quantitative IHC analysis of DRG neurons. If suboptimal tissue processing conditions are present, a shrinkage of DRG neuronal cytoplasm may result half-moon-shaped cytoplasmic morphology. Another form of imaging artifact is nuclear blebbing and is characterized as neurons with missing nuclei, resulting in a black hole where the nucleus should be, typically due to either perfusing too fast (> 37 mL/minute) or with insufficient volume (~100 mL/~50 grams) (Santoianni and Hammami, 2013; Petrilli et al., 2017). Other factors that affect tissue morphology include fixative composition, tissue type, cryostat section thickness, pH, and accidental negligence. Optimization of tissue processing *methods* is required if severe morphological skewing or loss occurs in either cytoplasmic or nuclear compartments.

Automated histologic image analysis is capable of efficiently measuring hundreds of images, increasing technique standardization between users and labs, while also reducing undetected error and saving considerable time. In fact, the estimation of total time required to produce regions of interest between each technique revealed an efficiency increase of ~24-fold, from manual to IC-DRG processing. Currently, the temporal bottleneck for measuring DRG neurons is the manual tracing step; however, an accurate method of automated image analysis would transfer the temporal bottleneck to image capturing. To determine an estimate minimum number of images required to yield a robust sample size, the total number of images in the naïve group was standardized to 29 images per naive rat (n=4), and the SD, SEM, and BCa confidence intervals were measured. There was a decreasing trend of SD as the number of images increased, as reflected by data collected from 5 images (SD = 5.02), 10 images (SD = 5.82), 15 images (SD = 4.72), 20 images (SD = 3.87), 25 images (SD = 3.32), and 29 images (SD = 2.92). The difference of SD between data collected from 20 images and data collected from 29 images was less than 1 SD value ($\Delta SD = 0.951$), illustrating minimal differences of SD from data collected in 20 to 29 images per group. In addition, SEM plotted from data measured in 2 – 29 images indicate that NeuN IR in DRG neuronal cytoplasm from 20 images, per group, is reflective of data measured from 29 images, per group (**Figure 3.12; *1**). BCa confidence intervals were computed to adjust for bias and skewness that may exist in the spread of bootstrap estimates. Most data points, representing mean data from 20 – 29 images, fall within 95% BCa confidence intervals, illustrating the sample size of NeuN IR in DRG neuronal cytoplasm collected from 20 – 29 images represent the true mean of the population (**Figure 3.12; *2**). Combined, these data suggest that 20 images per rat ($n \geq 4$), consisting of mainly neuronal soma, provide a robust sample size.

The potential range of DRG neuronal cytoplasmic NeuN IR expression was evaluated by the inclusion of the model of inflammation, representing the peak swelling timepoint in an acute, pain-state, activation of DRG neurons. To confirm inflammatory impact, the right hind paw was

measured from each rat, between conditions, and the AIA group presented with a significant increase over the naïve group, 2.35-fold average, when compared to control rats. This considerable level of inflammation closely mimics acute arthritis, indicating the establishment of expected dynamic molecular alterations in the DRG neuronal soma (Woolf and Ma, 2007).

A statistical comparison of NeuN IR between each technique revealed no statistical difference in DRG neuronal cytoplasm between naïve and AIA conditions, except for naïve CGRP (+) and AIA CGRP (+) populations (Table 1). There were, however, no significant differences found between the same size populations, e.g., small vs small, indicating that comparative neuronal subpopulations are identified in each technique (**Figure 3.7**). While CGRP-positive DRG neurons are activated by the peak timepoint of the AIA model, the expression of NeuN is stably expressed and serves as an accurate biomarker tool for the IC-DRG algorithm. Human, decision-based, measurement and IC-DRG algorithmic processing utilizes different methods of identifying and measuring the same objects. Accordingly, the expected mean between techniques should be similar, but not exact. Pearson coefficients were calculated to measure the linear correlation between NeuN IR in DRG neuronal cytoplasm between techniques and conditions. While each technique utilizes different methods of identifying and demarcating DRG neurons for measurement, both techniques are significantly correlated, for each condition. Collectively, these results indicate that both techniques are measuring similar populations, demonstrating that IC-DRG processing is both accurate and efficient.

The IC-DRG algorithm is designed to exploit the DRG neuronal cytoplasmic labeling pattern of NeuN IR and generate binary masks with automatically assigned demarcations of DRG neuronal nuclei and somata (Schindelin et al., 2012; Anderson, 2018). Accuracy depends upon strict size, shape, and pixel-intensity filters, discarding neuronal objects that lack proper dimensions. Furthermore, the code is designed to qualify the immunolabeling pattern of a pan-nuclear reporter (DAPI) as *neuronal nuclei* if it is located within the boundary of the DRG neuronal cytoplasmic

reporter (NeuN). Automatically generated binary masks of nuclei, imported into CellProfiler, are processed as seed objects, from which a boundary, representing the outer cytoplasmic membrane, expands in all directions and terminates at the outer cytoplasmic membrane, based on imported somatic masks. Nuclear and somatic masks are generated by IC-DRG and imported into CellProfiler to measure proteins of interest in nuclear, cytoplasmic, and whole cell (somatic) DRG neuronal compartments. IC-DRG requires two filter channels for operation, providing two or more channels for measurement of proteins of interest, depending on available filters. A CellProfiler template file is included and configured to export the nuclear and cytoplasmic DRG neuronal demarcations, based solely on IC-DRG masks, to manually score for false positive error (**Figure 3.4, B**).

Total count and total percent error were evaluated in size and CGRP populations to determine if they were equally identified and processed with similar accuracy. Restriction thresholds designed to eliminate type 1 errors are integrated into IC-DRG, containing proof-reading capability, removing suboptimal targets before measurement, resulting in slightly fewer neurons than manual identification and higher accuracy, as reflected by a total loss of 62 neurons for both, naïve (n=4) and AIA (n=4), groups and accuracy of 93.81% (SD=1.782). The IC-DRG script contains sensitive watershed functionality for accurate segmentation of DRG neurons, although, occasionally, small (and/or medium) size neurons were clumped together without proper watershed segmentation, exceeded size and circularity thresholds, and were automatically rejected from further measurement, increasing accuracy. Rejected neuronal clumps appeared to consist of small and medium size random neurons. A loss of neurons from manual (hand tracing) to algorithmic techniques is expected, due to strategies for increasing accuracy, although losses in total count may be attenuated by capturing additional field of views. Within size populations, small and medium size neurons were manually scored for percent error at 5% and 4%, respectively, while large DRG neurons were manually scored at 10% error. This was due to

general differences in morphology between relatively compact small and medium size neurons, compared to that of large size neurons which often contain sprawling cytoplasmic membrane boundaries. To modify the script for large DRG neuron inclusivity, parameter boundary values were relaxed and resulted in less error for large size neurons. However, relaxing these values led to increased clumping of small and medium size neurons, a reduction of total count, and decreased accuracy. The resulting IC-DRG script is highly accurate for small and medium sized neurons, a population known to project C and Aδ fibers into epithelia and respond to nociceptive and inflammatory stimuli, and less accurate for large neurons. In addition, every object manually scored as erroneous was cropped and carefully evaluated for patterns in the error. Sources of error appear to be due to non-specific IR, morphologic artifacts, or a combination of both (**Figure 3.5**). And while IC-DRG processing is more accurate in neurons that typically respond to pain and inflammation, the 10% error in large neurons, a population responsible for light touch (under normal circumstances), appear to be non-specific (Ma et al., 2003; Ririe et al., 2008). The source of error across DRG neuronal subpopulations appear to be random and more images may be required when evaluating large DRG neurons with IC-DRG.

Manual hand tracing is a traditional and accepted technique for measuring DRG neurons, however, this method may result in user-defined variability in identifying DRG neuronal nuclear and cytoplasmic boundaries, as well as determining which neurons are to be considered appropriate for measurement (e.g., sectioned through center of neuron). IC-DRG processing, on the other hand, singularly utilizes DAPI fluorescence to determine whether a neuron was sectioned through the center, based on size and nuclear thresholds for DAPI emission in rat DRG neurons, qualifying the object as an accepted representation of DRG neuronal nuclei for CellProfiler seeding and measurement of DRG neuronal nuclear, cytoplasmic, and whole cell compartments.

The IC-DRG script was designed for images captured at 20x. I found this optimal in DRG neurons for measuring size, sampling mean, and co-localization of channels. The IC-DRG algorithm uses a 'set scale' function to define size thresholds for a specific resolution of objective (pixel/micron) and may be recalculated for other objectives. In DRG neurons, images captured with a 10x objective resulted in low resolution for sampling means and images captured with a 40x objective resulted in few neurons per image with high resolution. If a study required the evaluation of total population count (including biomarkers for subpopulations) or the evaluation of high-resolution intracellular measurements, 10x and 40x (or higher), would be appropriate, respectively. Modifying the current script for other objectives may include updating the set scale function, updating pixel intensity threshold values, and possibly removing large sections of the script and folder system. The IC-DRG script was designed for repetition (overlapping regions of pixel intensity) with strict thresholds (to remove error) and much of this (and the folder system) could greatly be reduced for conversion to 10x.

I found epifluorescence microscopy appropriate for evaluating protein changes in 12µm sections of DRG tissue. Image stacks collected by confocal microscopy may be consolidated into one image, via Z-stack processing, and processed with the IC-DRG algorithm (with appropriate scale and threshold updates). It should be noted that every image (step) of the stack may contain a small amount of background fluorescence and will be additive during Z-stack merging, resulting in increased non-specific background. The IC-DRG ImageJ script is fully notated and divided into organized sections for intuitive adaptation to measure other cells in different tissues. While the IC-DRG script is designed for DRG neurons, it can be adapted to identify any cell of interest with the use of a relatively stable cytoplasmic fluorescence reporter, specific to the cell of interest, and DAPI. For example, this script could be adapted to automatically identify and measure specific immune cells in any tissue, based on DAPI and available cluster determinant (CD) antibodies. Adapting the current IC-DRG script for immune cells should primarily involve

modifications to size and intensity filters, within the script. Future studies may include the evaluation of IC-DRG processing in thicker tissue sections (20-40 μm) with confocal microscopy and Z-stack merging at high resolution to evaluate discrete changes in protein in a fewer number of neurons (than 20x). Adaptation of the current script for high magnification may include updating the set scale value for the objective (multiple places in script), fine-tuning size (μm^2) parameters in the analyze particle functions, and pixel intensity thresholds. The official ImageJ forum (https://forum.image.sc/) has been a valuable resource for asking questions and learning about ImageJ scripting.

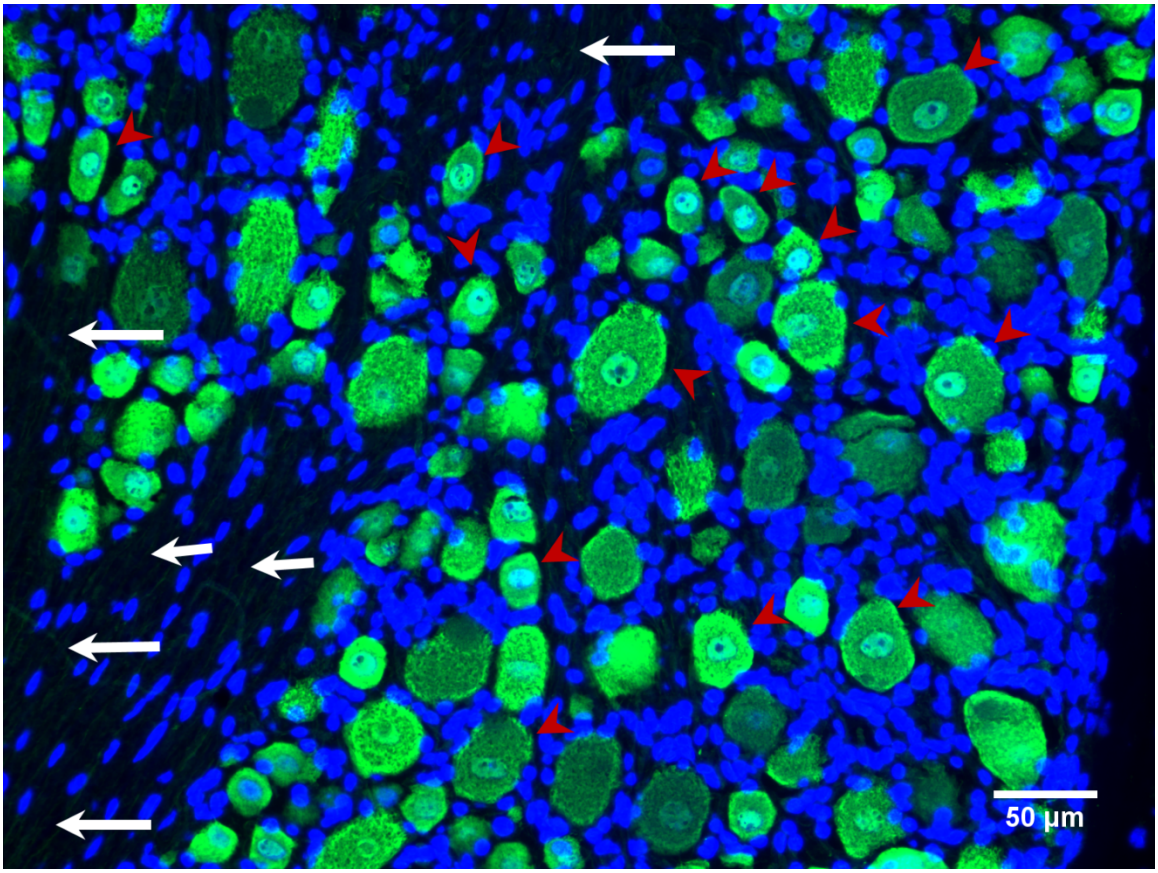
Identifying human error while manual hand tracing DRG neurons is difficult to track, locate, and correct. These types of errors primarily occur while determining which cells are suitable for tracing and in the comprehensive data organization steps. On the other hand, the IC-DRG script is an automated approach of neuronal identification and data collection that reduces human error and provides a quick and reliable method of evaluating total error per experiment. The established EMGI score system allows for microscope camera exposure calibration and standardization within, and among other, collaborating labs. IC-DRG processing provides users the freedom to increase experimental numbers per group, as well as total number of groups, per experiment without weeks of time-consuming manual tracing (Anderson, 2018). Comprehensively, these data illustrate that IC-DRG mask generation, in conjunction with CellProfiler for measurement, is an accurate and efficient method measuring neurons in DRG tissue with immunofluorescence.

Published August 2021

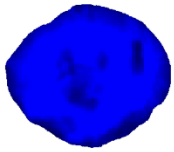
Analytical Biochemistry. Volume 627:114184, 15 August 2021

https://pubmed.ncbi.nlm.nih.gov/33811851

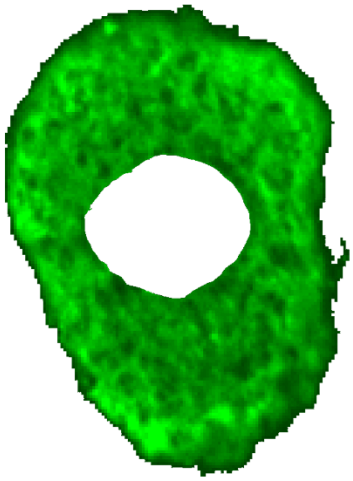
doi: doi.org/10.1016/j.ab.2021.114184


50 µm

A



B



C

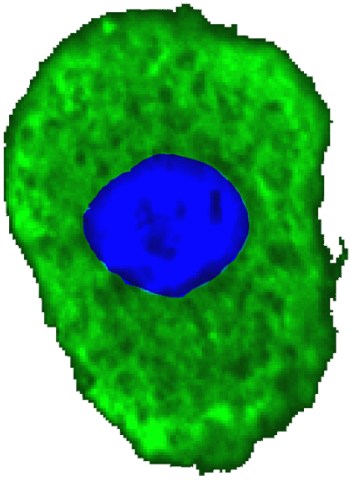
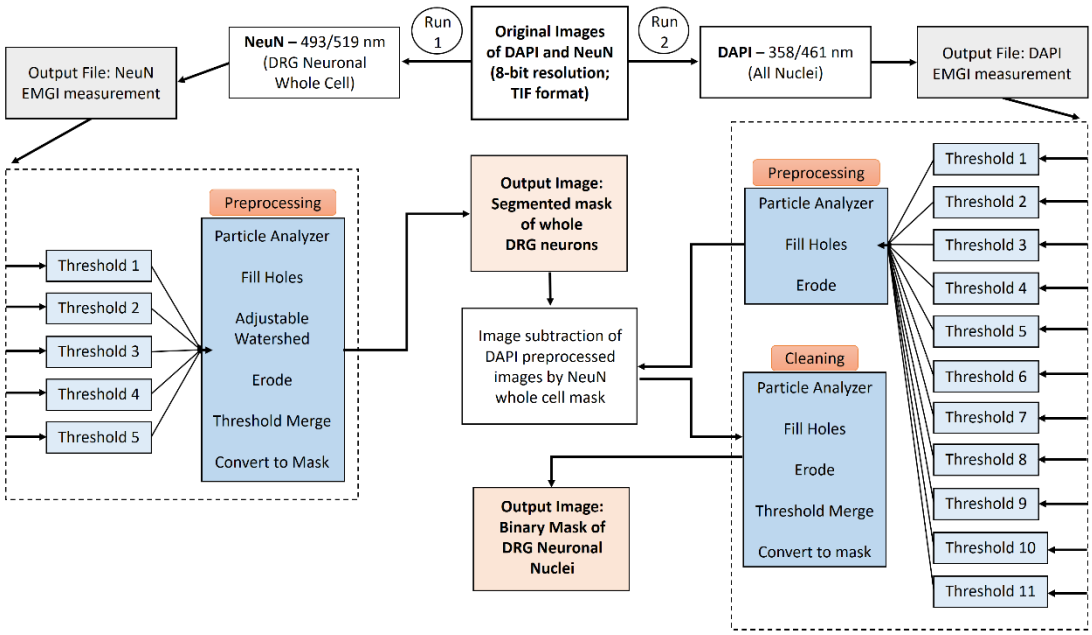
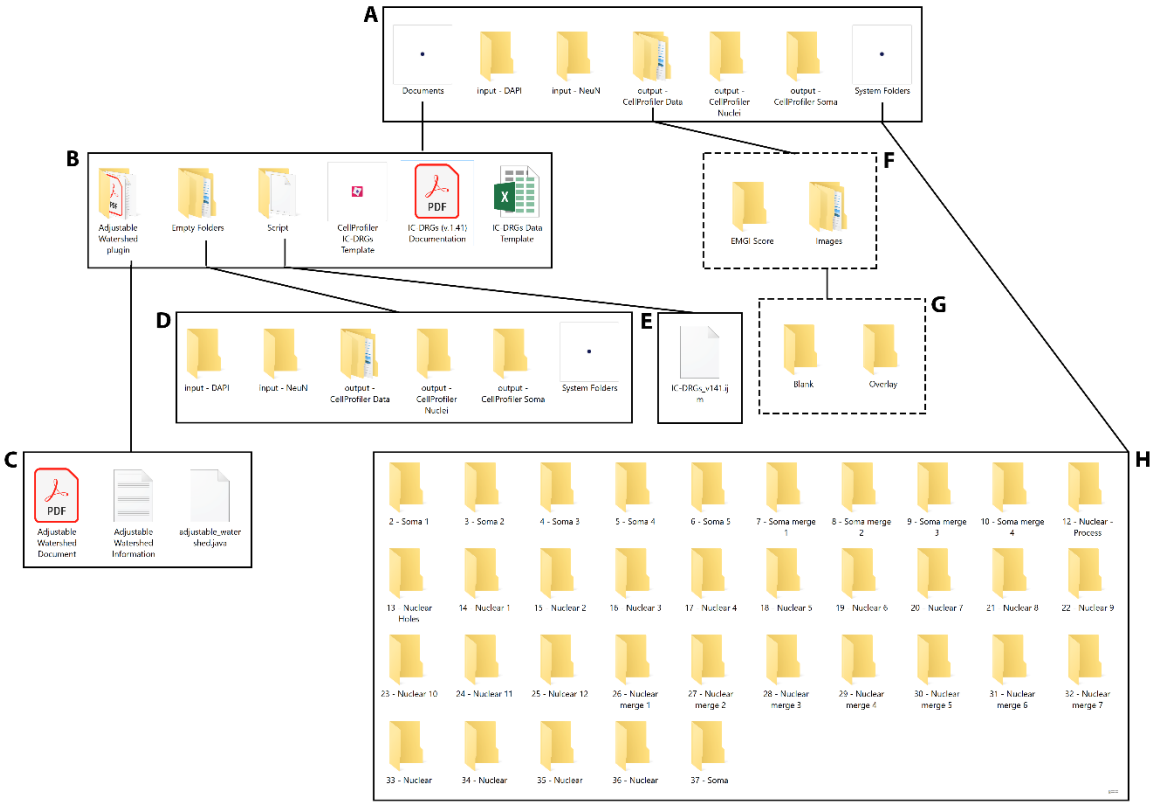
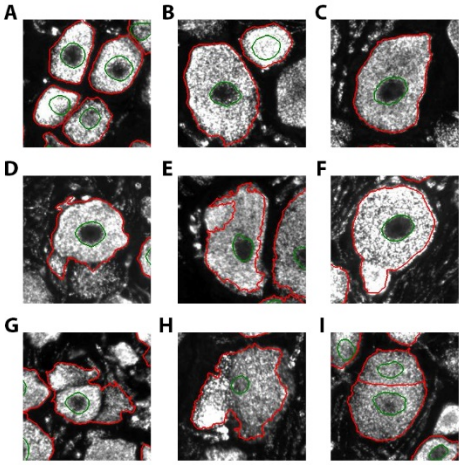
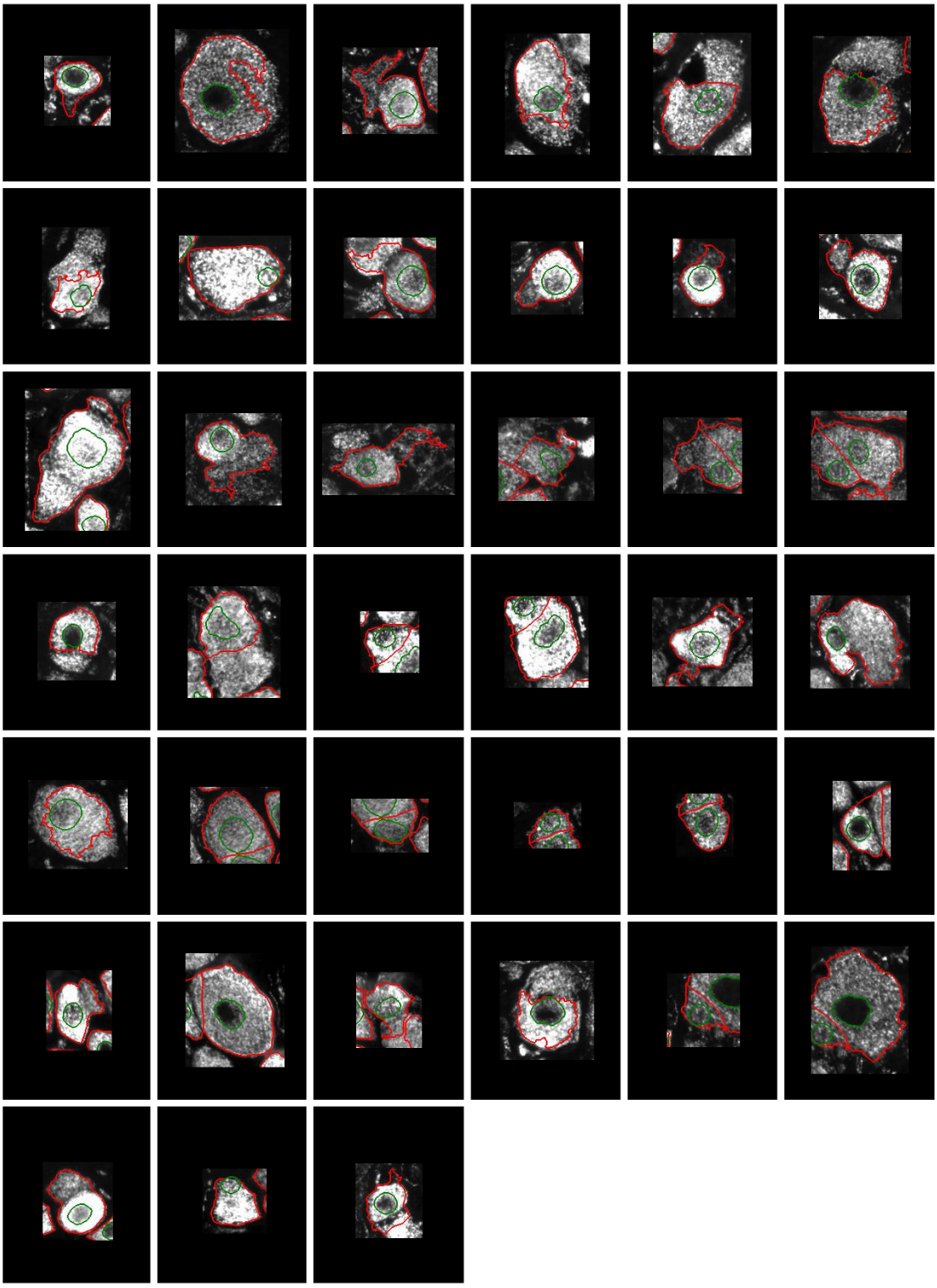


Figure 3.2 NeuN IR in DRG neuronal cytoplasmic and nuclear compartments. This image was created by manually removing nuclear NeuN IR (in ImageJ) and replacing it with its respective DAPI channel This manipulation was performed due to intense NeuN IR in DRG neuronal nuclei, reducing image contrast between neuronal sub compartments. **A**, neuronal nucleus; **B**, neuronal cytoplasm; **C**, neuronal whole cell (soma).


Output File: NeuN EMGI measurement
NeuN – 493/519 nm (DRG Neuronal Whole Cell)
Run 1
Original Images of DAPI and NeuN (8-bit resolution; TIF format)
Run 2
DAPI – 358/461 nm (All Nuclei)
Output File: DAPI EMGI measurement
Preprocessing
Particle Analyzer
Fill Holes
Adjustable Watershed
Erode
Threshold Merge
Convert to Mask
Threshold 1
Threshold 2
Threshold 3
Threshold 4
Threshold 5
Output Image: Segmented mask of whole DRG neurons
Image subtraction of DAPI preprocessed images by NeuN whole cell mask
Output Image: Binary Mask of DRG Neuronal Nuclei
Preprocessing
Particle Analyzer
Fill Holes
Erode
Cleaning
Particle Analyzer
Fill Holes
Erode
Threshold Merge
Convert to mask
Threshold 1
Threshold 2
Threshold 3
Threshold 4
Threshold 5
Threshold 6
Threshold 7
Threshold 8
Threshold 9
Threshold 10
Threshold 11


A
Documents
input - DAPI
input - NeuN
output - CellProfiler Data
output - CellProfiler Nuclei
output - CellProfiler Soma
System Folders
B
Adjustable Watershed plugin
Empty Folders
Script
CellProfiler IC-DRGs Template
IC DRGs (v 1.41) Documentation
IC DRGs Data Template
F
EMGI Score
Images
G
Blank
Overlay
D
input - DAPI
input - NeuN
output - CellProfiler Data
output - CellProfiler Nuclei
output - CellProfiler Soma
System Folders
E
IC-DRGs_v141.ijm
C
Adjustable Watershed Document
Adjustable Watershed Information
adjustable_watershed.java
H
2 - Soma 1
3 - Soma 2
4 - Soma 3
5 - Soma 4
6 - Soma 5
7 - Soma merge 1
8 - Soma merge 2
9 - Soma merge 3
10 - Soma merge 4
12 - Nuclear - Process
13 - Nuclear Holes
14 - Nuclear 1
15 - Nuclear 2
16 - Nuclear 3
17 - Nuclear 4
18 - Nuclear 5
19 - Nuclear 6
20 - Nuclear 7
21 - Nuclear 8
22 - Nuclear 9
23 - Nuclear 10
24 - Nuclear 11
25 - Nuclear 12
26 - Nuclear merge 1
27 - Nuclear merge 2
28 - Nuclear merge 3
29 - Nuclear merge 4
30 - Nuclear merge 5
31 - Nuclear merge 6
32 - Nuclear merge 7
33 - Nuclear
34 - Nuclear
35 - Nuclear
36 - Nuclear
37 - Soma


A
B
C
D
E
F
G
H
I



illustrates manually scored error is due to randomly dispersed fluorescence and morphological artifacts, resulting in erroneous nonspecific DRG boundary demarcation and type 1, false positive, error.

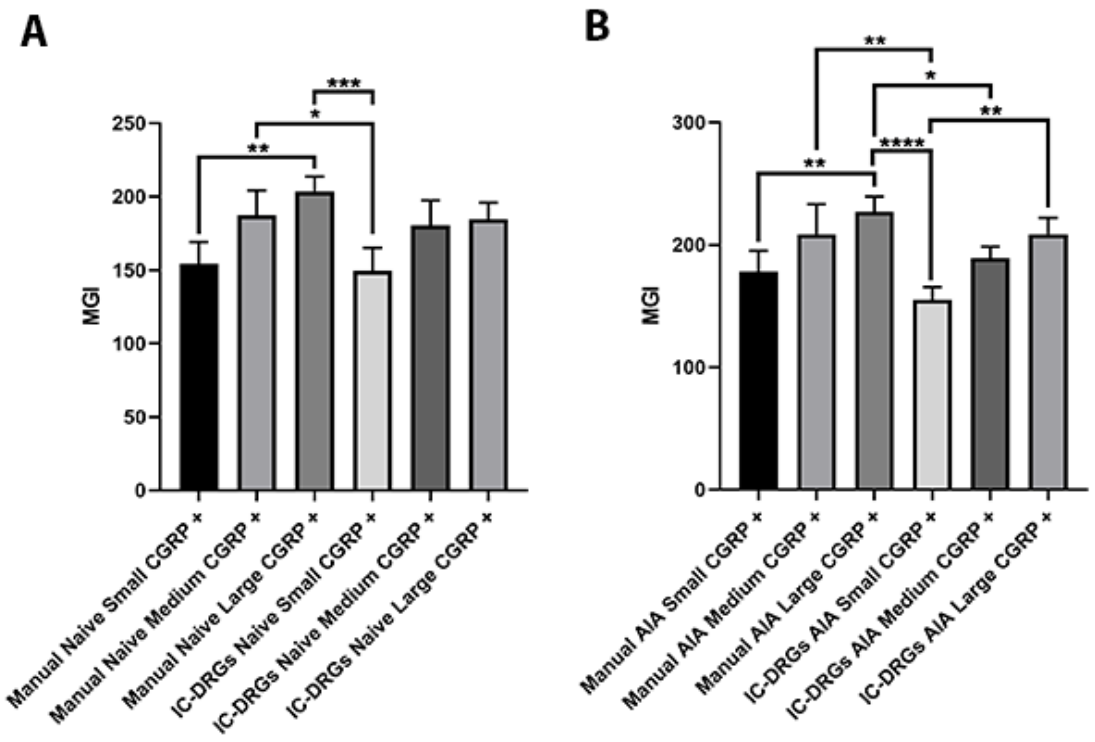


Figure 3.7 NeuN IR between techniques. NeuN IR in DRG neuronal cytoplasm between manual hand tracing and IC-DRG processing. There were no significant differences found between techniques within the same size population (e.g. small vs small) of NeuN IR in DRG neuronal cytoplasm**. A**, Significant differences found within size populations for naïve CGRP (+) cytoplasm ($F(5, 17) = 7.903$, $p = 0.0005$, pooled SD = 14.7). **B**, Significant differences found within size populations for AIA CGRP (+) ($F(5, 18) = 11.29$, $p < 0.0001$, pooled SD = 15.4). All data were tested for normality and statistically analyzed by one-way ANOVA, followed by Tukey post-hoc analysis. Column bar graphs are plotted as mean with SD.

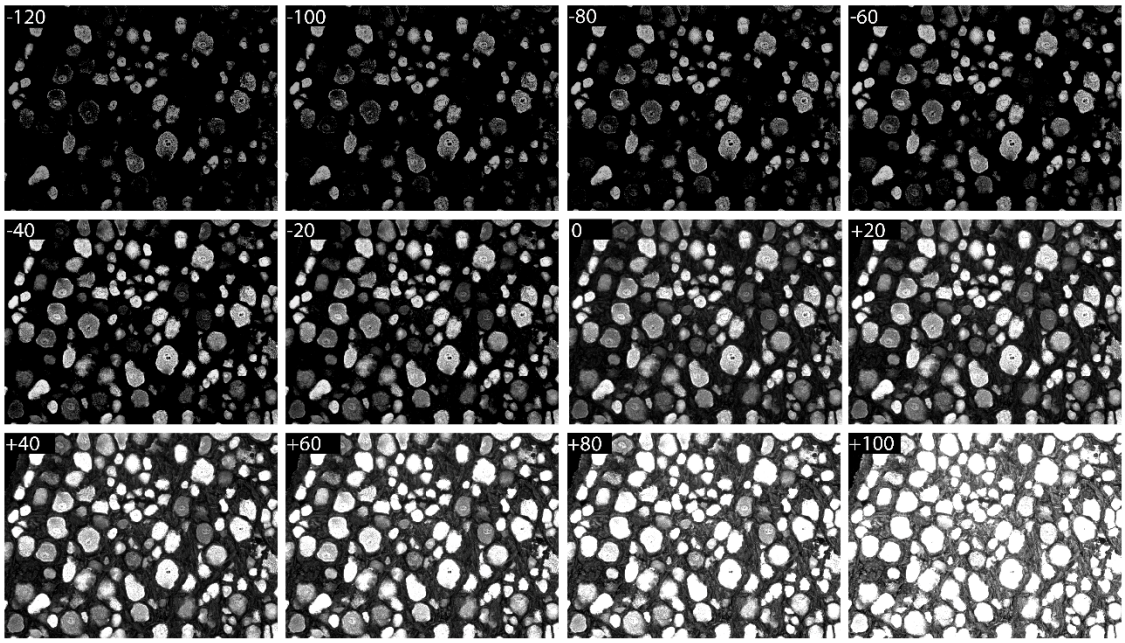
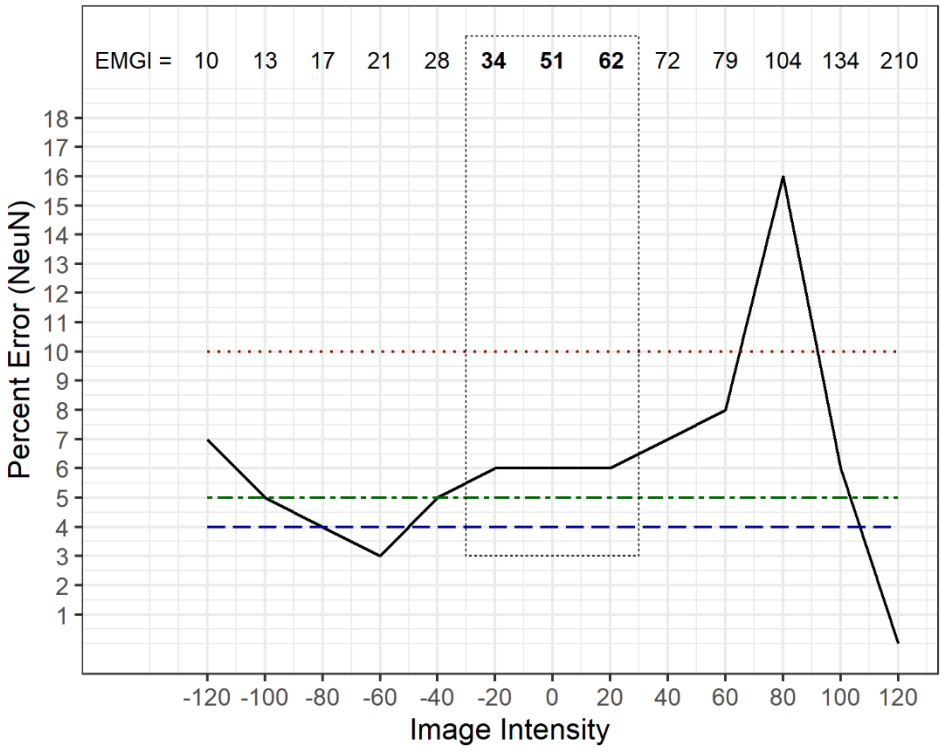


Figure 3.8 NeuN IR at different emulated camera exposures. Pixel intensity variation (PIV) script adjusted NeuN IR pixel brightness from the original pixel intensity (**VII**), in increments of 20 histogram levels (8-bit; 0–255 levels of pixel intensity), across 12 intensity variations.

A


EMGI = 10 13 17 21 28 34 51 62 72 79 104 134 210
Percent Error (NeuN)
18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
-120 -100 -80 -60 -40 -20 0 20 40 60 80 100 120
Image Intensity

B

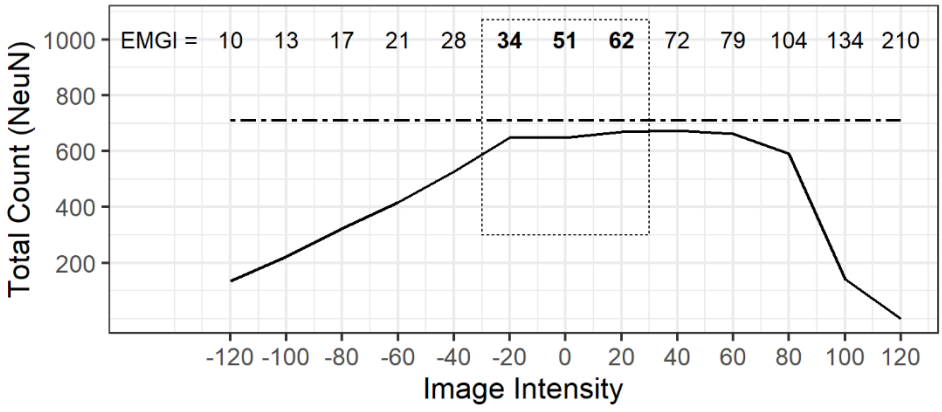
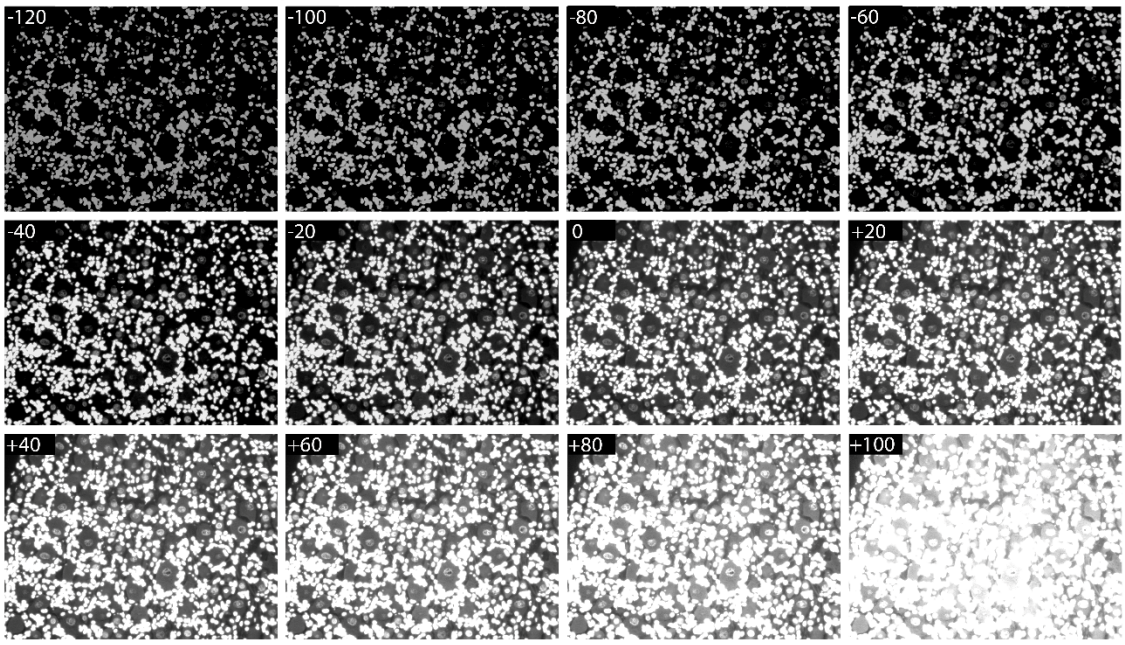
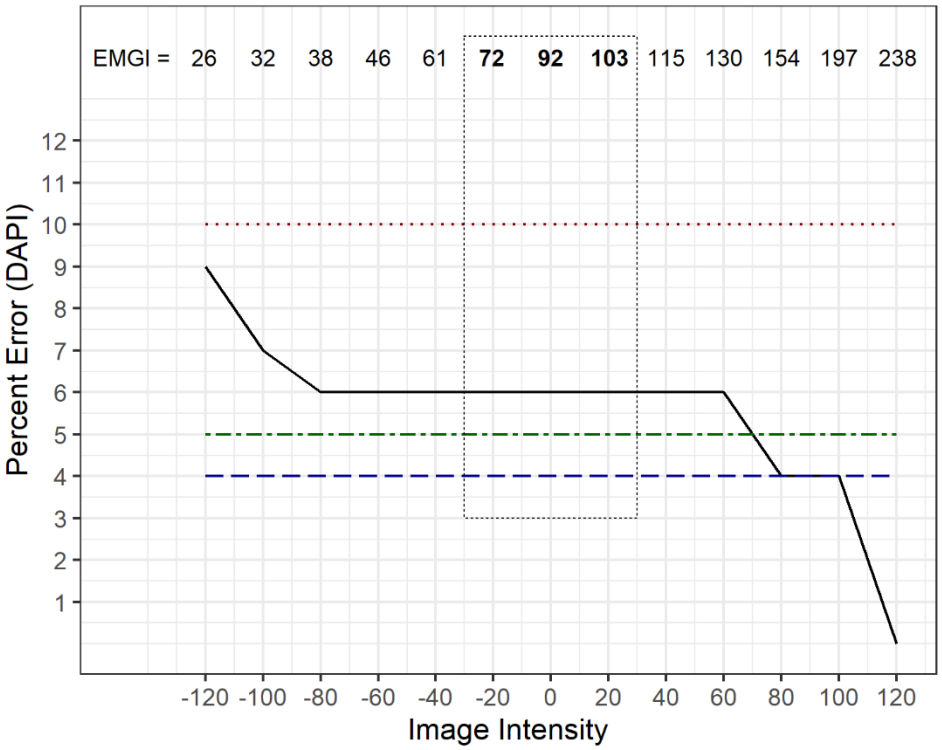

EMGI = 10 13 17 21 28 34 51 62 72 79 104 134 210
Total Count (NeuN)
1000 800 600 400 200
-120 -100 -80 -60 -40 -20 0 20 40 60 80 100 120
Image Intensity

Figure 3.9 NeuN IR-based EMGI chart. Comparison of NeuN IR EMGI score in reference of total count and manually scored error. **A,** manually scored error of neuronal objects in relation to camera exposure EMGI score. The solid black line is the total group, the blue dash horizontal line are small size neurons, the green two-dash horizontal line are medium size neurons, and the red dotted horizontal line are large size neurons. The EMGI score that corresponds with each image (pixel) intensity variation is at the top of the chart. The optimal range of EMGI scores that represent an area of low percent error is high-lighted in bold and is surrounded by a black dotted square. **B**, Total count of neuronal objects in relation to camera exposure EMGI score. The solid black line is the total count, identified by IC-DRG, across all groups of pixel intensity evaluation. The black two- dash horizontal line is the total count of the same set of images from manual hand tracing.


-120
-100
-80
-60
-40
-20
0
+20
+40
+60
+80
+100

A


EMGI = 26 32 38 46 61 72 92 103 115 130 154 197 238
Percent Error (DAPI)
12
11
10
9
8
7
6
5
4
3
2
1
-120 -100 -80 -60 -40 -20 0 20 40 60 80 100 120
Image Intensity

B

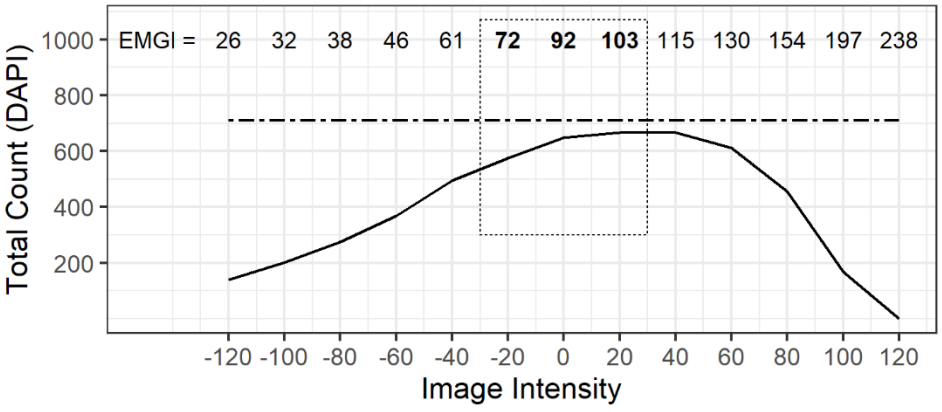
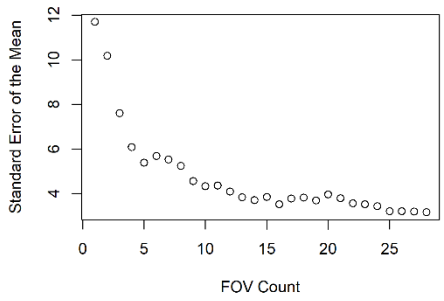
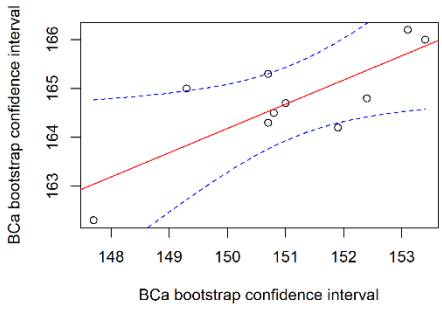

EMGI = 26 32 38 46 61 72 92 103 115 130 154 197 238
Total Count (DAPI)
1000
800
600
400
200
-120 -100 -80 -60 -40 -20 0 20 40 60 80 100 120
Image Intensity

Figure 3.11 DAPI IR-based EMGI chart. Comparison of DAPI EMGI score in reference of total count and manually scored error. **A**, manually scored error of neuronal objects in relation to camera exposure EMGI score. The solid black line is the total group, the blue dash horizontal line are small size neurons, the green two-dash horizontal line are medium size neurons, and the red dotted horizontal line are large size neurons. The EMGI score that corresponds with each image (pixel) intensity variation is at the top of the chart. The optimal range of EMGI scores that represent an area of low percent error is high-lighted in bold and is surrounded by a black dotted square. **B**, Total count of neuronal objects in relation to camera exposure EMGI score. The solid black line is the total count, identified by IC-DRG, across all groups of pixel intensity evaluation. The black two-dash horizontal line is the total count of the same set of images from manual hand tracing. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

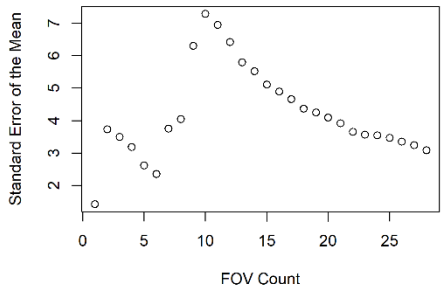
A1



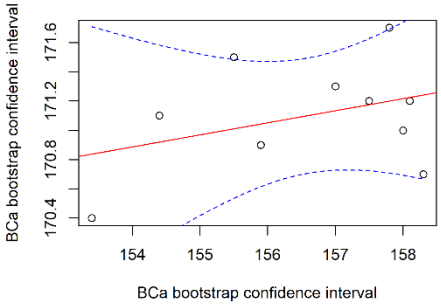
A2



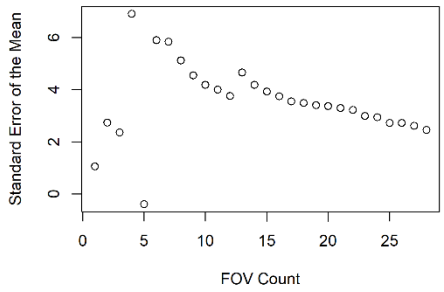
B1



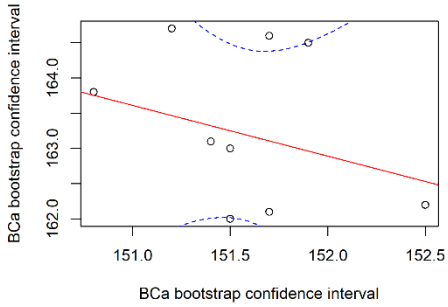
B2



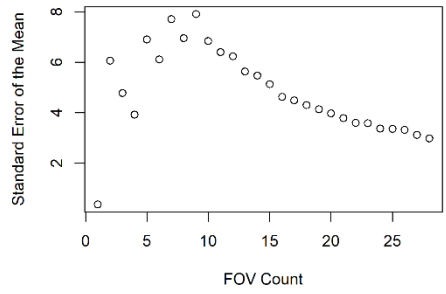
C1



C2



D1



D2

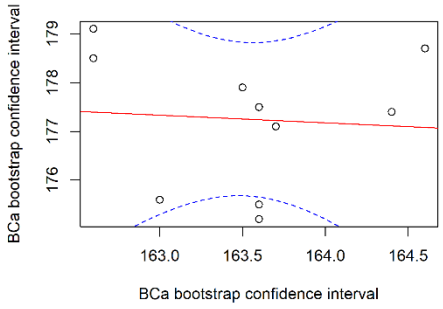


Figure 3.12 Confidence intervals of NeuN IR in DRG neuronal cytoplasm. Standard error of the mean (SEM) and bias-corrected and accelerated (BCa) 95% confidence intervals of NeuN IR in DRG neuronal cytoplasm. **A**, naïve rat 1; **B**, naïve rat 2; **C**, naïve rat 3; **D**, naïve rat 4; **A1/B1/C1/D1**, plot of SEM for every field of view for data collected from 2 to 29 fields of view (images). **A2/ B2/C2/D2**, plot of linear trend line with 95% BCa confidence intervals for data collected from 20 to 29 fields of view.

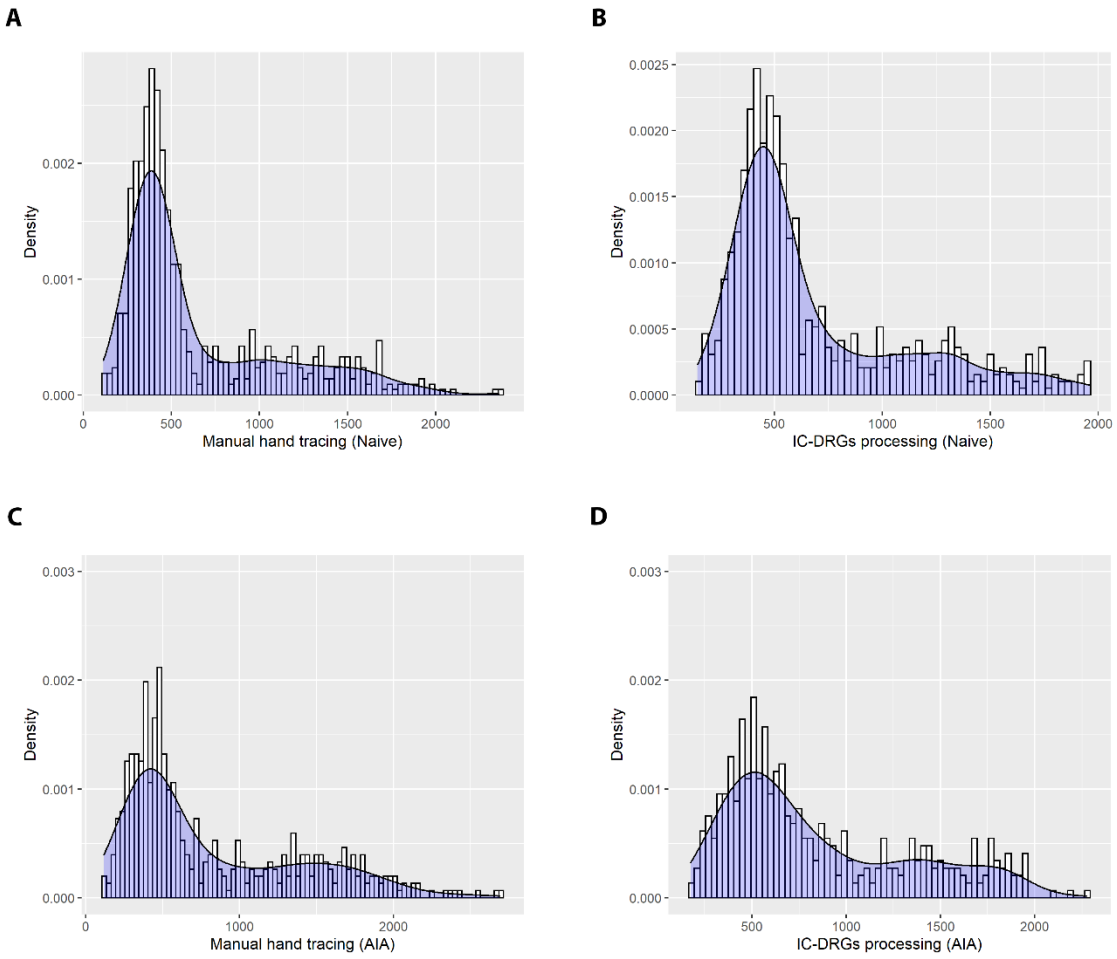


Figure 3.13 DRG neuronal size distribution. **A**, size distribution of manual hand tracing from the naïve group; **B**, size distribution of IC-DRG processing from the naïve group; **C**, size distribution of manual hand tracing from the inflammatory group; **D**, size distribution of IC-DRG processing from the inflammatory group.

Test details	Mean 1	Mean 2	Mean Diff.	95.00% CI of diff.	Significant?	Summary	Adjusted P Value	SE of diff.	n1	n2	q	DF
Manual Naive Small CGRP + vs. Manual Naive Medium CGRP +	154.3	187.3	-33.01	-66.19 to 0.1608	No	ns	0.0515	10.37	4	4	4.502	17
Manual Naive Small CGRP + vs. Manual Naive Large CGRP +	154.3	203.4	-49.12	-82.29 to -15.94	Yes	**	0.0022	10.37	4	4	6.698	17
Manual Naive Small CGRP + vs. IC-DRGs Naive Small CGRP +	154.3	149.6	4.754	-28.42 to 37.93	No	ns	0.997	10.37	4	4	0.6482	17
Manual Naive Small CGRP + vs. IC-DRGs Naive Medium CGRP +	154.3	180.4	-26.1	-59.28 to 7.073	No	ns	0.1736	10.37	4	4	3.559	17
Manual Naive Small CGRP + vs. IC-DRGs Naive Large CGRP +	154.3	184.7	-30.33	-66.16 to 5.506	No	ns	0.1249	11.2	4	3	3.829	17
Manual Naive Medium CGRP + vs. Manual Naive Large CGRP +	187.3	203.4	-16.1	-49.28 to 17.07	No	ns	0.6376	10.37	4	4	2.196	17
Manual Naive Medium CGRP + vs. IC-DRGs Naive Small CGRP +	187.3	149.6	37.77	4.593 to 70.94	Yes	*	0.0208	10.37	4	4	5.15	17
Manual Naive Medium CGRP + vs. IC-DRGs Naive Medium CGRP +	187.3	180.4	6.912	-26.26 to 40.09	No	ns	0.9834	10.37	4	4	0.9426	17
Manual Naive Medium CGRP + vs. IC-DRGs Naive Large CGRP +	187.3	184.7	2.687	-33.15 to 38.52	No	ns	0.9999	11.2	4	3	0.3392	17
Manual Naive Large CGRP + vs. IC-DRGs Naive Small CGRP +	203.4	149.6	53.87	20.70 to 87.04	Yes	***	0.0009	10.37	4	4	7.346	17
Manual Naive Large CGRP + vs. IC-DRGs Naive Medium CGRP +	203.4	180.4	23.02	-10.16 to 56.19	No	ns	0.2793	10.37	4	4	3.138	17
Manual Naive Large CGRP + vs. IC-DRGs Naive Large CGRP +	203.4	184.7	18.79	-17.04 to 54.62	No	ns	0.5631	11.2	4	3	2.372	17
IC-DRGs Naive Small CGRP + vs. IC-DRGs Naive Medium CGRP +	149.6	180.4	-30.86	-64.03 to 2.319	No	ns	0.0766	10.37	4	4	4.207	17
IC-DRGs Naive Small CGRP + vs. IC-DRGs Naive Large CGRP +	149.6	184.7	-35.08	-70.91 to 0.7521	No	ns	0.0569	11.2	4	3	4.429	17
IC-DRGs Naive Medium CGRP + vs. IC-DRGs Naive Large CGRP +	180.4	184.7	-4.225	-40.06 to 31.61	No	ns	0.9988	11.2	4	3	0.5334	17
Manual AIA Small CGRP + vs. Manual AIA Medium CGRP +	177.9	208.8	-30.9	-65.43 to 3.628	No	ns	0.0951	10.86	4	4	4.022	18
Manual AIA Small CGRP + vs. Manual AIA Large CGRP +	177.9	227.2	-49.29	-83.82 to -14.76	Yes	**	0.003	10.86	4	4	6.416	18
Manual AIA Small CGRP + vs. IC-DRGs AIA Small CGRP +	177.9	155.1	22.82	-11.71 to 57.34	No	ns	0.3302	10.86	4	4	2.97	18
Manual AIA Small CGRP + vs. IC-DRGs AIA Medium CGRP +	177.9	189.4	-11.5	-46.03 to 23.03	No	ns	0.8912	10.86	4	4	1.497	18
Manual AIA Small CGRP + vs. IC-DRGs AIA Large CGRP +	177.9	208.8	-30.86	-65.39 to 3.670	No	ns	0.0957	10.86	4	4	4.017	18
Manual AIA Medium CGRP + vs. Manual AIA Large CGRP +	208.8	227.2	-18.39	-52.92 to 16.14	No	ns	0.5532	10.86	4	4	2.394	18
Manual AIA Medium CGRP + vs. IC-DRGs AIA Small CGRP +	208.8	155.1	53.72	19.19 to 88.24	Yes	**	0.0013	10.86	4	4	6.992	18
Manual AIA Medium CGRP + vs. IC-DRGs AIA Medium CGRP +	208.8	189.4	19.4	-15.13 to 53.93	No	ns	0.4983	10.86	4	4	2.525	18
Manual AIA Medium CGRP + vs. IC-DRGs AIA Large CGRP +	208.8	208.8	0.04144	-34.49 to 34.57	No	ns	>0.9999	10.86	4	4	0.0054	18
Manual AIA Large CGRP + vs. IC-DRGs AIA Small CGRP +	227.2	155.1	72.11	37.58 to 106.6	Yes	****	<0.0001	10.86	4	4	9.386	18
Manual AIA Large CGRP + vs. IC-DRGs AIA Medium CGRP +	227.2	189.4	37.79	3.260 to 72.32	Yes	*	0.0273	10.86	4	4	4.919	18
Manual AIA Large CGRP + vs. IC-DRGs AIA Large CGRP +	227.2	208.8	18.43	-16.10 to 52.96	No	ns	0.5509	10.86	4	4	2.399	18
IC-DRGs AIA Small CGRP + vs. IC-DRGs AIA Medium CGRP +	155.1	189.4	-34.32	-68.85 to 0.2096	No	ns	0.0519	10.86	4	4	4.467	18
IC-DRGs AIA Small CGRP + vs. IC-DRGs AIA Large CGRP +	155.1	208.8	-53.67	-88.20 to -19.15	Yes	**	0.0013	10.86	4	4	6.987	18
IC-DRGs AIA Medium CGRP + vs. IC-DRGs AIA Large CGRP +	189.4	208.8	-19.36	-53.88 to 15.17	No	ns	0.5005	10.86	4	4	2.52	18

Table 3.1 NeuN IR in all DRG neuronal populations. NeuN IR in small, medium, large, CGRP-positive, and CGRP-negative DRG neuronal populations.

Category	Total Counts	Total Errors	Percent Error in population
Total identified (small)	153	8	5.23
Total identified (medium)	316	13	4.11
Total identified (large)	179	18	10.06
Total identified (CGRP +)	222	13	5.86
Total identified (CGRP -)	426	26	6.10
Total identified	648	39	6.02

Table 3.2 Manually scored error in size and CGRP subpopulations.

CHAPTER IV

Separation of rat epidermis and dermis with thermolysin to detect site-specific inflammatory mRNA and protein

Summary

Presented here is a protocol for the separation of epidermis from dermis to evaluate inflammatory mediator production. Following inflammation, rat hind paw epidermis is separated from the dermis by thermolysin at 4 °C. The epidermis is then used for mRNA analysis by RT-PCR and protein evaluation by western blot and immunohistochemistry.

Abstract

Easy-to-use and inexpensive techniques are needed to determine the site specific production of inflammatory mediators and neurotrophins during skin injury, inflammation, and/or sensitization. The goal of this study is to describe an epidermal/dermal separation protocol using thermolysin, a proteinase that is active at 4 °C.

To illustrate this procedure, Sprague Dawley rats are anesthetized, and right hind paws are injected with carrageenan. Six and twelve hours after injection, rats with inflammation and naïve rats are euthanized, and a piece of hind paw, glabrous skin is placed in cold Dulbecco's Modified Eagle Medium. The epidermis is then separated at the basement membrane from the dermis by thermolysin in PBS with calcium chloride. Next, the dermis is secured by microdissection forceps, and the epidermis is

gently teased away. Toluidine blue staining of tissue sections show that the epidermis is separated cleanly from the dermis at the basement membrane. All keratinocyte cell layers remain intact, and the epidermal rete ridges along with indentations from dermal papillae are clearly observed. Qualitative and real-time RT-PCR is used to determine nerve growth factor and interleukin-6 expression levels. Western blotting and immunohistochemistry are finally performed to detect amounts of nerve growth factor. This report illustrates that cold thermolysin digestion is an effective method to separate epidermis from dermis for evaluation of mRNA and protein alterations during inflammation.

Introduction

Evaluation of inflammatory mediators and neurotrophic factors from the skin can be limited due to the heterogeneity of cell types found in the inflamed dermis and epidermis (Schakel et al., 2016; Manti et al., 2017; Choi and Di Nardo, 2018). Several enzymes, chemical, thermal, or mechanical techniques involving separation of the two layers or for performing cell dissociation for evaluation have been reviewed recently (Zou and Maibach, 2018). Acid, alkali, neutral salt, and heat can divide the epidermis from dermis quickly, but cellular and extracellular swelling often occurs (Hu et al., 2006). Trypsin, pancreatin, elastase, keratinase, collagenase, pronase, dispase, and thermolysin are enzymes that have been used for epidermal-dermal separation (Einbinder et al., 1966; Zou and Maibach, 2018). Trypsin and other broad scale proteolytic enzymes are active at 37–40 °C and must be monitored carefully to prevent dissociation of epidermal layers. Dispase cleaves the epidermis at the lamina densa, and requires 24 h for separation in the cold (Rakhorst et al., 2006; Zou and Maibach, 2018) or shorter timepoints at 37 °C (Tschachler et al., 2004; Zou and Maibach, 2018). A limiting feature of all these techniques is the potential disruption of tissue morphology and loss of integrity of mRNA and protein.

To maintain the integrity of mRNA and protein, a skin separation method should be carried out in the cold for a short period of time. In evaluating skin separation techniques for inflammation studies, thermolysin is an effective enzyme to separate the epidermis from dermis at cold temperatures (Zou and Maibach, 2018). Thermolysin is active at 4 °C, cleaves epidermal hemidesmosomes from the lamina lucida, and separates the epidermis from dermis within 1–3 h (Walzer et al., 1989; Rakhorst et al., 2006; Zou and Maibach, 2018). The goal of this report is to optimize the use of thermolysin for separation of inflamed rat epidermis from dermis to detect mRNA and protein levels for inflammatory mediators and neurotrophic factors. Several preliminary reports have been presented (Anderson et al., 2010; Ibitokun et al., 2010; Nawani et al., 2011; Gujar and Miller, 2017; Anderson and Miller, 2018a). The objective of this manuscript is to describe an optimal skin separation technique using thermolysin and demonstrate the detection of 1) markers of inflammation, 2) interleukin-6 (IL-6) mRNA, and 3) nerve growth factor (NGF) mRNA and protein in the epidermis of rats with carrageenan-induced inflammation (C-II) (Fehrenbacher et al., 2012). A preliminary report using the complete Freund's adjuvant model indicates that NGF mRNA and protein levels increase early during inflammation (Gujar and Miller, 2017). In mice, skin sensitization with the topical application of oxazolone causes an early rise in the IL-6 mRNA using in situ hybridization (Flint et al., 1998). Both IL-6 and NGF have been implicated in carrageenan induced inflammation (C-II) (Li et al., 2018), but there have been no reports describing mRNA or protein levels for IL-6 or NGF, specifically from the epidermis during the acute stage of C-II.

The thermolysin technique is inexpensive and straightforward to perform. Furthermore, thermolysin separation of the epidermis from dermis allows for mRNA, western blot, and immunohistochemical analysis of inflammatory mediators and neurotrophic factors during the process of inflammation (Gujar and Miller, 2017). Investigators should be able to easily use this technique in both preclinical and clinical studies of skin inflammation.

Methods

This protocol follows the animal care guidelines of Oklahoma State University Center for Health Sciences IACUC (#2016-03).

1. Carrageenan-induced inflammation (C-II)

1. Anesthetize male and/or female Sprague Dawley rats (200–250 g; 8–9 weeks old) with isoflurane (or injectable anesthetic).
2. Check the depth of anesthesia by touching the cornea and lightly pinching the left hind paw. When the animal is appropriately anesthetized, no corneal or paw response will be observed.
3. Subcutaneously inject the right glabrous, hind paw with 100 µL of 1% (w/v) λ-carrageenan diluted in phosphate-buffered saline (PBS) (Hoffman et al., 2010).
    1. Make sure that appropriate controls are used, such as naïve rats without isoflurane in this report. Preliminary studies indicate that naïve rats with or without isoflurane have the same basal expression of epidermal IL-6 and NGF. NOTE: Naïve rats are preferred controls for inflammation studies since subcutaneous saline or PBS cause a local inflammation (Hoffman and Miller, 2010; Crosby et al., 2015; Hoffman et al., 2016).
4. Evaluate the edema of C-II rats to guarantee the effectiveness of the carrageenan (**Figure 4.1**) (Hoffman and Miller, 2010; Hoffman et al., 2010). Determine the amount of edema by measuring the hind paw metatarsal thickness with calipers.
5. At 6–12 h, euthanize rats with CO_2 (or injectable anesthetic overdose) and cut 1 mm x 2 mm pieces of glabrous hind paw skin with sharp scalpel. If hairy skin is used, then shave it before cutting the 1 mm x 2 mm pieces of skin.

NOTE: Make sure that the appropriate timepoints are chosen according to the specific studies.

6. Using microdissection forceps, transfer the skin into 1 mL of cold Dulbecco's Modified Eagle Medium (DMEM) in a microcentrifuge tube on ice and keep cold for 15–60 min.

2. Thermolysin separation of epidermis and dermis

1. Prepare and activate thermolysin.
    1. Prepare a solution of thermolysin, by adding 5 mg of *Geobacillus stearothermophilus* to 10 mL of PBS, at pH = 8 (concentration 500 μg/mL).
    2. Prepare a 1 M solution of calcium chloride ($CaCl_2$ anhydrous) by adding 1.11 g into 10 mL of distilled H_2O.
    3. To prevent autolysis of thermolysin, add 10 μL of calcium chloride to 10 mL of thermolysin solution. The calcium chloride final concentration will be 1 mM.
    4. Aliquot 1 mL of activated thermolysin into 10 wells of a 24 well cell culture plate on ice.
2. Use thermolysin enzyme digestion to separate the epidermis from dermis.
    1. Using microdissection forceps, transfer one skin sample into each well of activated thermolysin. Make sure not to immerse the skin in the thermolysin solution.
    2. Gently tap the skin on the side of the well to assist in releasing the skin sample from the forceps to float on the thermolysin solution.
    3. Float the skin into the thermolysin solution with the *stratum corneum* (outer epidermis) side up and dermis facing down. It is critical that the dermis faces down, or the effective separation will not take place.

       NOTE: The amount of time for thermolysin incubation must be determined

empirically by the end-user. Glabrous, hind paw skin from Sprague Dawley rats (200–250 g; 8–9 weeks old) often requires 2.0–2.5 h for separation. Incubation time is expected to vary with species and age.

4. After the appropriate incubation time in thermolysin, use microdissection forceps to transfer one skin sample into a well of a 6 well cell culture plate with 7–8 mL of cold (4 °C) DMEM. This allows more room for separation of the epidermis from the dermis.
5. Immerse the skin into the DMEM.
6. Gently brush the epidermis with the forceps around the perimeter of the skin until the near-translucent epidermis is observed at the borders. If this cannot be achieved, return the skin sample to the thermolysin solution for another 15–30 min.
7. Once the epidermis noticeably separates from the dermis, then carefully hold both the epidermis and dermis with microdissection forceps and very slowly pull the epidermis from the dermis.
8. Evaluate the translucence of the isolated epidermis and make sure it is optically consistent. See **Figure 4.2** for an example of a 1 mm x 2 mm sample of rat epidermis. If there is a variation in the translucence, then proper separation has not occurred.

3. Inactivate thermolysin using ethylenediaminetetraacetic acid (EDTA) in the separated pieces of epidermis and dermis.

   CAUTION: The thermolysin that remains in the epidermis and dermis is still active and can damage the layers if not inactivated.

   1. Prepare a 0.5 M EDTA stock solution. To do so, slowly add 0.93 g EDTA into 5 mL of double-distilled water. Add sodium hydroxide to the solution until it clears. Ensure that the pH of the solution is ~8.0.

2. Make a 5 mM EDTA solution in DMEM. Add 0.25 mL of 0.5 M EDTA stock solution to 25 mL of DMEM.
3. Place the separated epidermis and dermis into the 5 mM EDTA/DMEM solution at 4 °C for 30 min to deactivate thermolysin's activity.

4. Evaluate the epidermis with tinctorial histology (Walzer et al., 1989; Tschachler et al., 2004; Rakhorst et al., 2006).
    1. Fix a portion of the epidermis in a 10% neutral formalin, 4% paraformaldehyde, or 0.25% paraformaldehyde with 0.8% picric acid solution for 1 h at room temperature (RT) with agitation.
    2. Place the fixed epidermis in 10% sucrose in PBS for 1 hour at RT with agitation.
    3. Freeze the epidermis in a tissue embedding matrix for sectioning. Cut 14 μm cross-sections using a cryostat and thaw-mount sections onto gelatin-coated glass microscope slides.
    4. Dry sections on a slide warmer and stain with a working solution of toluidine blue (TB; 10% TB in 1% sodium chloride) for 90 s. Appose coverslips with an aqueous mounting medium.
    5. Observe the epidermis with brightfield microscopy a 10x-20x objective.

       NOTE: If proper separation has occurred, the epidermis will be divided cleanly from the dermis and the five layers will be detected: *stratum basale*, *stratum spinosum*, *stratum granulosum*, *stratum lucidum,* and *stratum corneum*. An example of separated rat skin epidermis can be seen in **Figure 4.3**.

3. Protein extraction and western blot analysis

1. Perform western blotting on the separated tissue samples using previously published protocol (Crosby et al., 2015).

2. Homogenize the epidermis in 50 µL of lysis buffer (25mM Tris HCl, pH = 7.4, 150 mM NaCl, 1 mM EDTA, 5% glycerol, and 1% Triton X-100) containing a phosphatase and protease inhibitor cocktail.
3. Centrifuge samples at a max speed for 15 min at 4 °C and evaluate the supernatant for protein concentration using a protein assay kit.
4. Load equal concentrations of protein (30 µg) onto SDS gels, perform electrophoresis, and then transfer proteins to nitrocellulose or PVDF membranes.
5. Block membranes with 5% milk for 2 h and incubate overnight in primary antibody (mouse anti-NGF, E12, 1:1000).
6. Wash 3x with PBS with 0.3% tween for 10 min each and incubate with a labeled secondary antibody (e.g., alkaline phosphatase labeled rabbit anti-mouse IgG).
7. Use a scanning system to evaluate western blot signal (e.g., ECF substrate and an imaging platform).

4. Immunohistochemistry

1. Place tissue samples in a fixative for optimal immunoreactivity: 0.96% (w/v) picric acid and 0.2% (w/v) formaldehyde in 0.1 M sodium phosphate buffer, pH = 7.3[21,22,23] for 4 h at RT. Transfer to 10% sucrose in PBS overnight at 4 °C.
2. Perform standard immunohistochemistry on the tissue sections (Burg et al., 2004; Hoffman et al., 2010; Hoffman et al., 2016).
3. Embed the epidermis from animals into a single frozen block in embedding matrix and cut 10–30 µm sections on a cryostat. Mount the sections on gelatin-coated, glass microscope slides and dry at 37 °C for 2 h.
4. Wash sections for three, 10 min rinses in PBS and incubate for 24–96 h in primary antisera, [e.g., mouse anti-NGF (E12, 1:2000)] and rabbit anti-protein gene product 9.9

(PGP 9.5, 1:2000) diluted in PBS containing 0.3% (w/v) Triton X-100 (PBS-T) PBS-T with 0.5% bovine serum albumin (BSA) and 0.5% polyvinylpyrrolidone (PVP).

5. After primary antiserum incubation, rinse sections three times for 10 min in PBS and incubate 1 h at RT in Alexa Fluor 488 donkey anti-rabbit IgG (1:1000) and Alexa Fluor 555 donkey anti-mouse IgG (1:1000) diluted in PBS-T.
6. Rinse sections three times in PBS for 10 min and affix coverslips with non-fading mounting medium to retard fading of immunofluorescence.

5. RNA isolation and cDNA synthesis

1. Perform standard reverse transcriptase polymerase chain reaction (RT-PCR) on the skin samples[21]. Isolate total RNA using a phenol, guanidine isothiocyanate solution.
2. Carry out complementary DNA synthesis by Moloney murine leukemia virus reverse transcriptase.
3. Use the following primer sequences for NGF and IL-6 amplification:
   NGF (Sense) - GTGGACCCCAAACTGTTTAAGAAACGG
   NGF (Antisense) – GTGAGTCCTGTTGAAGGAGATTGTACCATG
   IL-6 (Sense) - GCAATTCTGATTGTATGAACAGCGATGATGC;
   IL-6 (Antisense) – GTAGAAACGGAACTCCAGAAGACCAGAG
4. Compare the levels of NGF and IL-6 mRNA to β-actin housekeeping gene:
   β-ACTIN (Sense) - TGCGTGACATTAAAGAGAAGCTGTGCTATG
   β-ACTIN (Antisense) – GAACCGCTCATTGCCGATAGTGATGA
5. Evaluate with qualitative RT-PCR using thermal cycler and quantitative real-time PCR (qRT-PCR) using a qRT-PCR system.

Results

Carrageenan injection into the rat hind paw caused classic symptoms of inflammation such as redness and edema (Hoffman and Miller, 2010; Fehrenbacher et al., 2012). The swelling of the hind paw was measured with mechanical calipers (Hoffman and Miller, 2010). A baseline value of the thickness of the paw was obtained for each rat before carrageenan treatment and measured again at 6 h and 12 h. Paw thickness was increased significantly compared to the baseline values (**Figure 4.1**).

Thermolysin incubation of the rat glabrous hind paw skin produced a sheet of epidermis. Brightfield microscopy was used to evaluate the effectiveness of thermolysin separation of the epidermis and dermis (**Figure 4.2**). The layers of the epidermis could be determined while focusing through the sheet at higher magnification. Toluidine blue staining of epidermal cross sections showed that the epidermis was separated from the dermis at the basement membrane (**Figure 4.3**). The epidermal rete ridges (epidermal pegs) along with the indentations from dermal papillae were intact. All keratinocyte cell layers were observed.

Western blotting of thermolysin-separated epidermis produced consistent results indicating stable protein levels during the technique at 4 °C. Very little NGF protein was detected in naïve rat epidermis, but NGF protein levels were upregulated (250%) after 6 h of C-II as compared to naïve animals (**Figure 4.4**). After 12 h of C-II, NGF levels were reduced compared to 6 h but remained elevated (55%) relative to controls. Immunohistochemistry for NGF in the separated epidermis provided reliable immunostaining and confirmed the results from western blots (**Figure 4.5**). NGF-immunoreactivity (ir) was not detected in naïve control epidermis, but at 6 h C-II, there was NGF-ir in most of the keratinocytes of the stratum granulosum and stratum lucidum. A few cells for the stratum spinosum were NGF-immunoreactive (IR) at 6 h C-II. At 12 h C-II, NGF-ir occurred in keratinocytes of the stratum granulosum and stratum lucidum with

some cells in stratum granulosum intensely NGF-IR. At no timepoint was NGF-ir detected in stratum basale or stratum corneum. PGP9.5-IR intraepidermal, varicose nerve fibers were present in the separated epidermis from naïve and C-II rats (**Figure 4.5**).

Qualitative RT-PCR demonstrated that there was good quality mRNA from thermolysin-separated epidermis (**Figure 4.6, A**; **Figure 4.7, A**). Using actin as a housekeeping gene for quantitative real-time PCR, NGF mRNA expression in epidermis during C-II was significantly elevated (>3-fold) at 6 h compared to naïve rats (**Figure 4.6, B**). At 12 h, NGF mRNA returned to baseline levels (**Figure 4.6, B**). Using actin as a housekeeping gene for quantitative real-time PCR, IL-6 mRNA in epidermis during C-II was significantly elevated (>6-fold) at 6 h compared to naïve rats (**Figure 4.7, B**). At 12 h, IL-6 mRNA levels dropped significantly from the 6 h amounts but remained elevated (2-fold) compared to naïve rats (**Figure 4.7, B**).

Discussion

The study determined that the epidermis of rat hind paw glabrous skin was easily separated from dermis using thermolysin (0.5 mG/mL) in PBS with 1 mM calcium chloride at 4 °C for 2.5 h. Histological evaluation indicated that the epidermis was separated from the dermis at the basement membrane and that the epidermal rete ridges were intact. Thermolysin is an extracellular metalloendopeptidase produced by Gram-positive (*Geo*)*Bacillus thermoproteolyticus* (Walzer et al., 1989; Burg et al., 2013). Its activity is stable at 4 °C but is functional over a wide range of temperatures (Walzer et al., 1989; Burg et al., 2013). This enzyme has been used extensively for protein chemistry (Kresge et al., 2009; Burg et al., 2013), however, several groups have shown its application for skin separation in epidermal and/or dermal sheets (Walzer et al., 1989; Glade et al., 1996; Hybbinette et al., 1999; Rakhorst et al., 2006; Zou and Maibach, 2018). Walzer et al. were the first to report epidermal-dermal separation of human skin using thermolysin at 4 °C (250–500 μg/mL for 1 h) (Walzer et al., 1989). With

light and electron microscopy, separation was determined to occur at the epidermal basement membrane between laminin and the bullous pemphigoid antigen site (Walzer et al., 1989).

Furthermore, hemidesmosomes, the attachments of basal keratinocytes to the basement membrane, were disrupted selectively (Walzer et al., 1989; Gragnani et al., 2007). Rakhorst et al. compared thermolysin (4 °C, 500 μg/mL, overnight) to dispase for epidermal-dermal separation of rabbit buccal mucosa (Rakhorst et al., 2006). Thermolysin was incomplete in separating the mucosal epidermis from dermis signifying that differences may occur for species, incubation time, solution composition (no $CaCl_2$ to prevent thermolysin autolysis), source of thermolysin, and/or site-specific differences indicated from other studies (Walzer et al., 1989; Glade et al., 1996; Hybbinette et al., 1999). End users of the current protocol should make sure to use fresh thermolysin and always include calcium chloride but also should be aware of these potential limitations.

Although some investigators have used thermolysin at 37 °C (Hybbinette et al., 1999; Gragnani et al., 2007), users of this protocol should be mindful to keep skin tissue at 4 °C to preserve the stability of protein and mRNA. I used DMEM at 4 °C as a solution for skin prior to and after thermolysin separation because of its usefulness in maintaining cells in culture (Sato and Kan, 2001), and it has been used previously for skin separation with thermolysin (Glade et al., 1996; Hybbinette et al., 1999; Michel et al., 1999; Rakhorst et al., 2006). However, Walzer et al. used sterile PBS supplemented with 200 μG/mL streptomycin, 200 U/mL penicillin, and 2.5 μg/mL fungizone (Walzer et al., 1989), whereas others have used different media (e.g., keratinocyte culture media free of epidermal growth factor) followed by PBS rinsing (Gragnani et al., 2007).

In the protocol, thermolysin separation was performed in 500 μg/mL thermolysin and 5 mM calcium chloride in PBS (pH = 8), similar to the original method (Walzer et al., 1989). DMEM has been used as a solution for thermolysin separation at 4 °C (overnight) (Rakhorst et al., 2006),

and HEPES buffer has been used effectively with 500 μG/mL thermolysin solution at 37 °C for 2 hours (Gragnani et al., 2007). However, I did not explore how culture medium or other buffers affects thermolysin's activity for epidermal-dermal separation. Calcium chloride is an important addition to decrease autolysis of thermolysin (Fassina et al., 1986; Burg et al., 2013) and deletion of this step may lead to incomplete cleavage of the epidermis from dermis (Rakhorst et al., 2006).

The size of the skin sample appears to influence the time needed for thermolysin incubation and the effectiveness of enzymatic cleavage of the epidermis from dermis. Investigators need to evaluate the appropriate sample size and incubation time for their own tissues. Floating the samples on the thermolysin solution with the epidermis facing upward is important for optimal enzyme effectiveness (Walzer et al., 1989). As noted earlier, the site of action for thermolysin is at the keratinocyte hemidesmosomes and basement membrane (Walzer et al., 1989; Gragnani et al., 2007; Zou and Maibach, 2018); therefore, thermolysin works inward from the edges of the skin. From our experience, the skin edges separate earlier than the middle of the sample, and it is important to allow enough time for complete enzymatic cleavage. When cleavage is complete, the epidermis should pull away easily from the dermis. If still attached, tugging on the layers may cause portions of dermis to come away with the epidermis.

A limitation of the thermolysin technique is the time required. Increasing the thermolysin concentration beyond 500 μG/mL does not decrease the time for separation and there is poor preservation of the epidermis at higher concentrations (Walzer et al., 1989). Epidermal-dermal separation methods have been reviewed recently (Zou and Maibach, 2018), and many methods take 30–60 min at 20–40 °C. Heat (50–60 °C) separation of skin occurs quickly (30 s to 10 min) (Zou and Maibach, 2018), but proteins and mRNA are known to degrade quickly at such high temperatures. Alternatively, sodium thiocyanate (2 N) at RT may be an acceptable rapid separation technique (5 min)(Felsher., 1947; Zou and Maibach, 2018), but protein and mRNA integrity have not been studied with this method (Felsher., 1947). The cold thermolysin method

was chosen for the preservation of protein and mRNA, but there were no direct comparisons made between protein and mRNA integrity using other techniques.

In the present study, cold thermolysin digestion is demonstrated to be an effective method to separate the epidermis from the dermis for evaluation of mRNA and protein alterations during inflammation. During carrageenan-induced inflammation, NGF mRNA and protein levels and IL-6 mRNA levels were elevated at 6 h, returning close to baseline by 12 h. With immunohistochemistry, NGF immunoreactivity was increased in keratinocytes at 6 h and 12 h. An advantage of the thermolysin method is the ability to perform site-selective analysis. For example, the increased production of NGF and IL-6 in the current study is from keratinocytes, since dermal cells are excluded from the assays. This method allows for insight into the location and mediator types for sensitization of primary afferent terminals (Petho and Reeh, 2012). In addition, this method allows for better understanding of the time course of neurotrophin production along with uptake and transport in primary afferents during inflammation (Djouhri et al., 2001; Denk et al., 2017).

Published September 2021

Journal of Online Video Experimentation

https://www.jove.com/v/59708/separation-rat-epidermis-dermis-with-thermolysin-to-detect-site

doi: 10.3791/59708

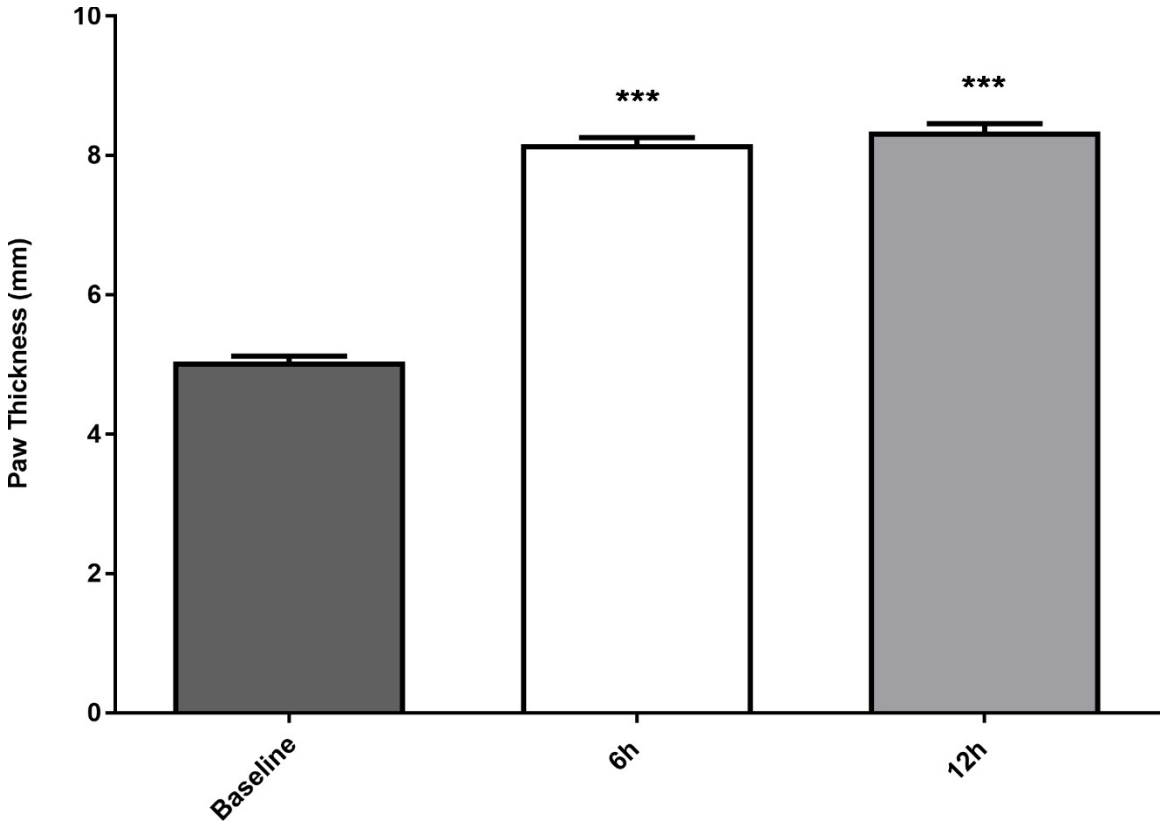


Figure 4.1: **Carrageenan injection produces paw edema.** A baseline value was obtained for each rat prior to carrageenan treatment. After 6 h and 12 h of treatment, paw thickness increased significantly compared to baseline values. The results are expressed as the SEM with six rats per treatment (*$p < 0.05$, **$p < 0.01$, ***$p < 0.001$; student's t-test, unpaired, two-tail, was performed at each timepoint).

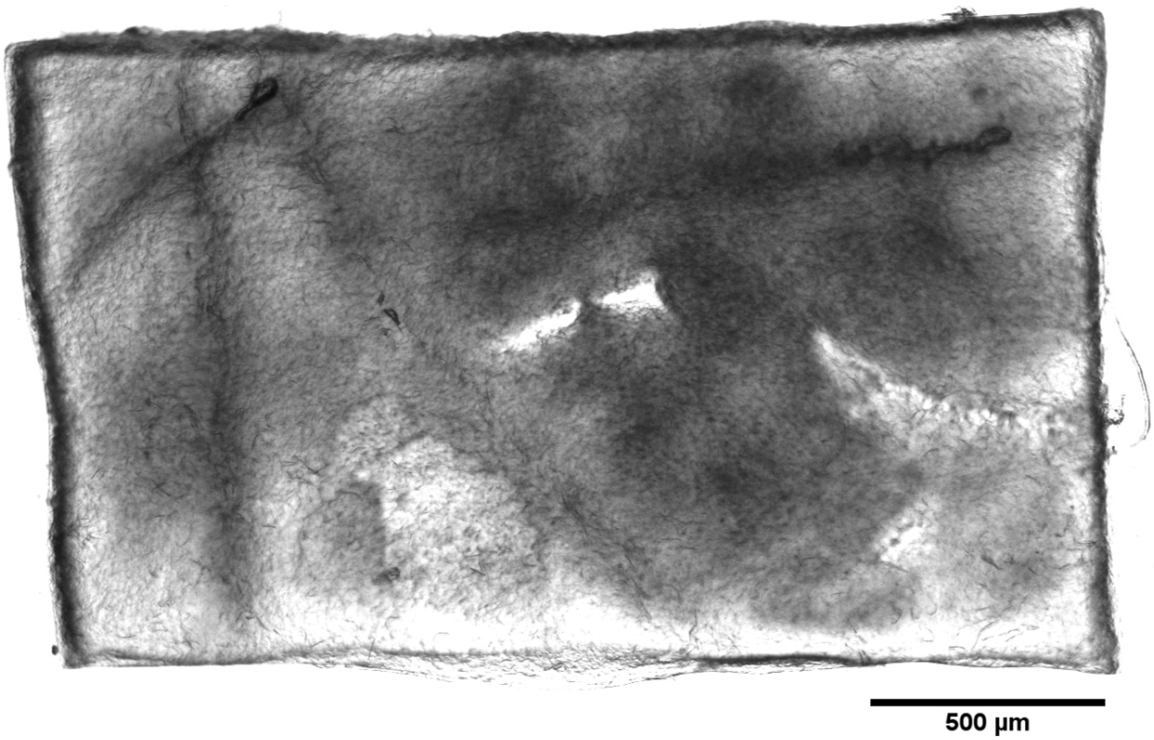


Figure 4.2 Thermolysin processed epidermis. Brightfield microscopy revealed a translucent epidermal sheet approximately 1 mm x 2 mm in size. The layers of the epidermis could be determined while focusing through the sheet at higher magnification. Scale bar = 500 µm.

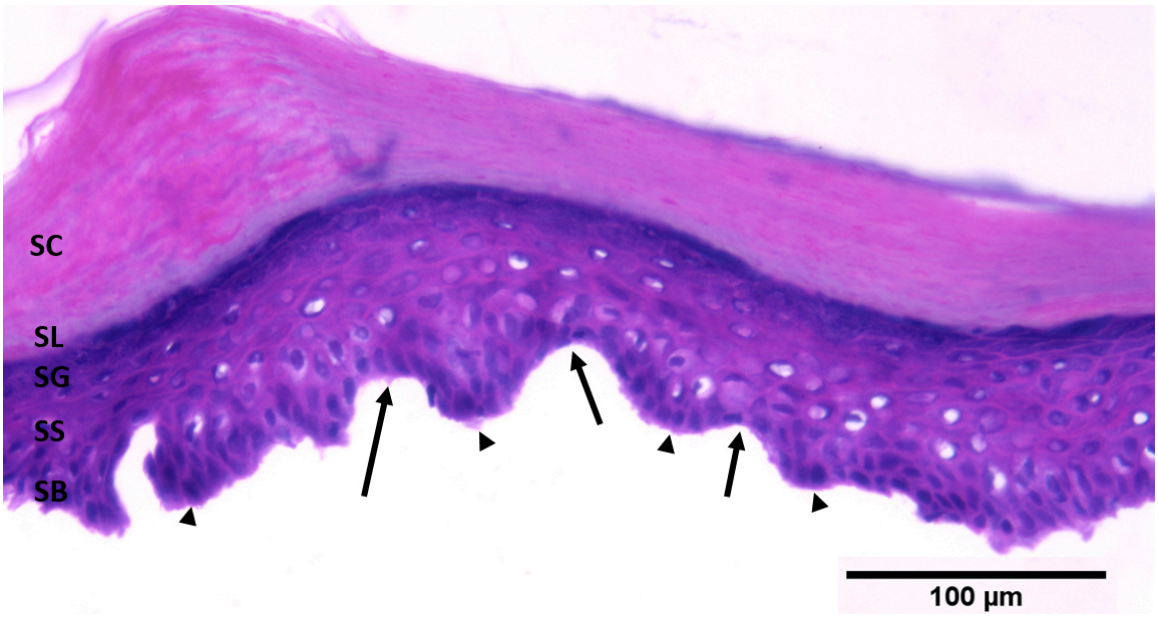


Figure 4.3 Toluidine blue staining in the epidermis. Brightfield microscopy determined that thermolysin caused an effective separation of the epidermis from dermis. Epidermal rete ridges (epidermal pegs; arrowheads) were observed along with indentations from dermal papillae (arrows). All keratinocyte cell layers were intact. SB: stratum basale, SS: stratum spinosum, SG: stratum granulosum, SL: stratum lucidum, SC: stratum corneum. Scale bar = 100 µm.

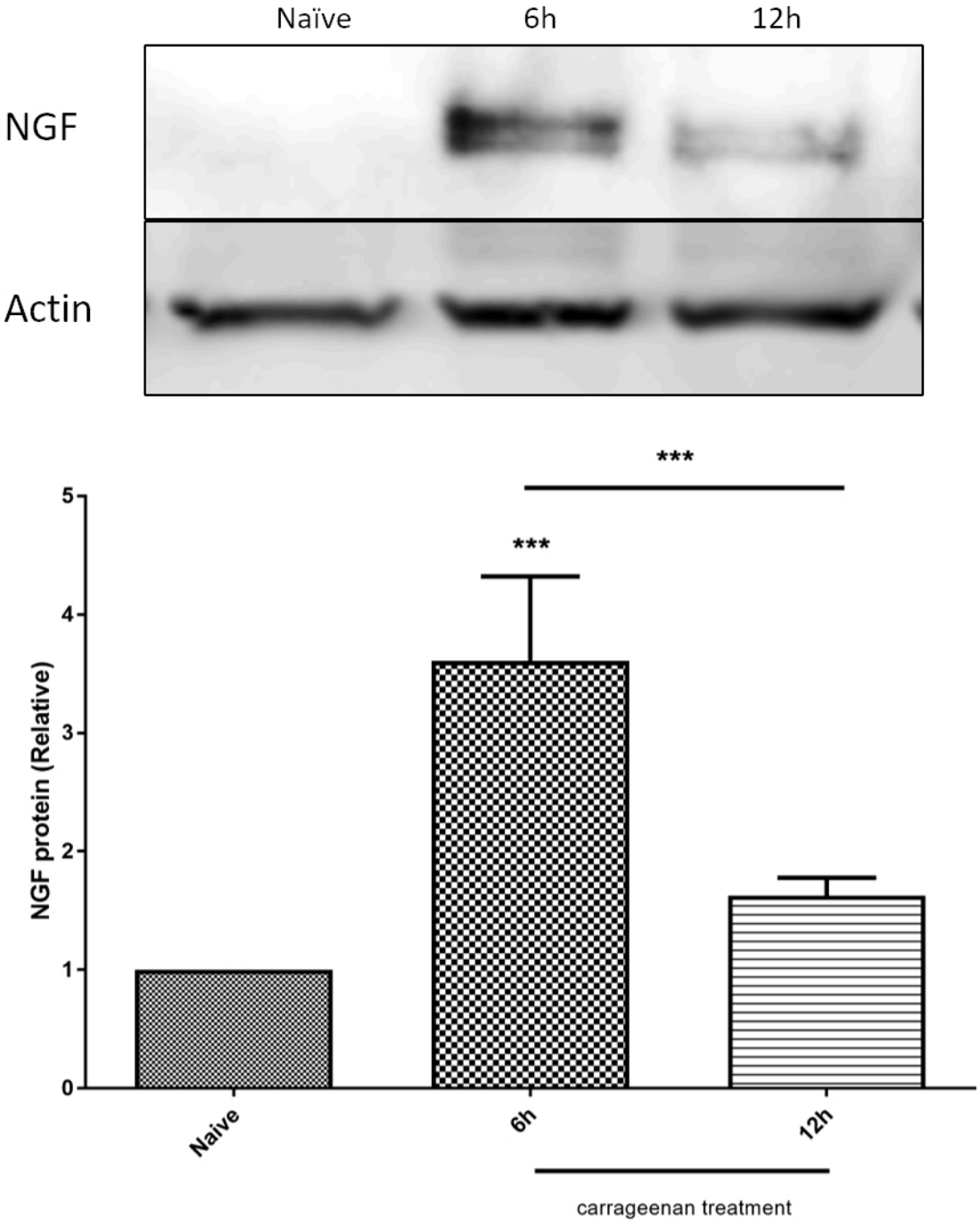


Figure 4.4 NGF protein expression during carrageenan. NGF protein expression in epidermis during carrageenan-induced inflammation. NGF levels were increased (250%) after 6 h of inflammation compared to naïve animals. After 12 h, the levels were reduced compared to 6 h but were elevated (55%) in contrast to naïve animals. The results are expressed as the SEM with three

rats per group (*$p < 0.05$, **$p < 0.01$, ***$p < 0.001$; student's t-test, unpaired, two-tail was performed at each timepoint).

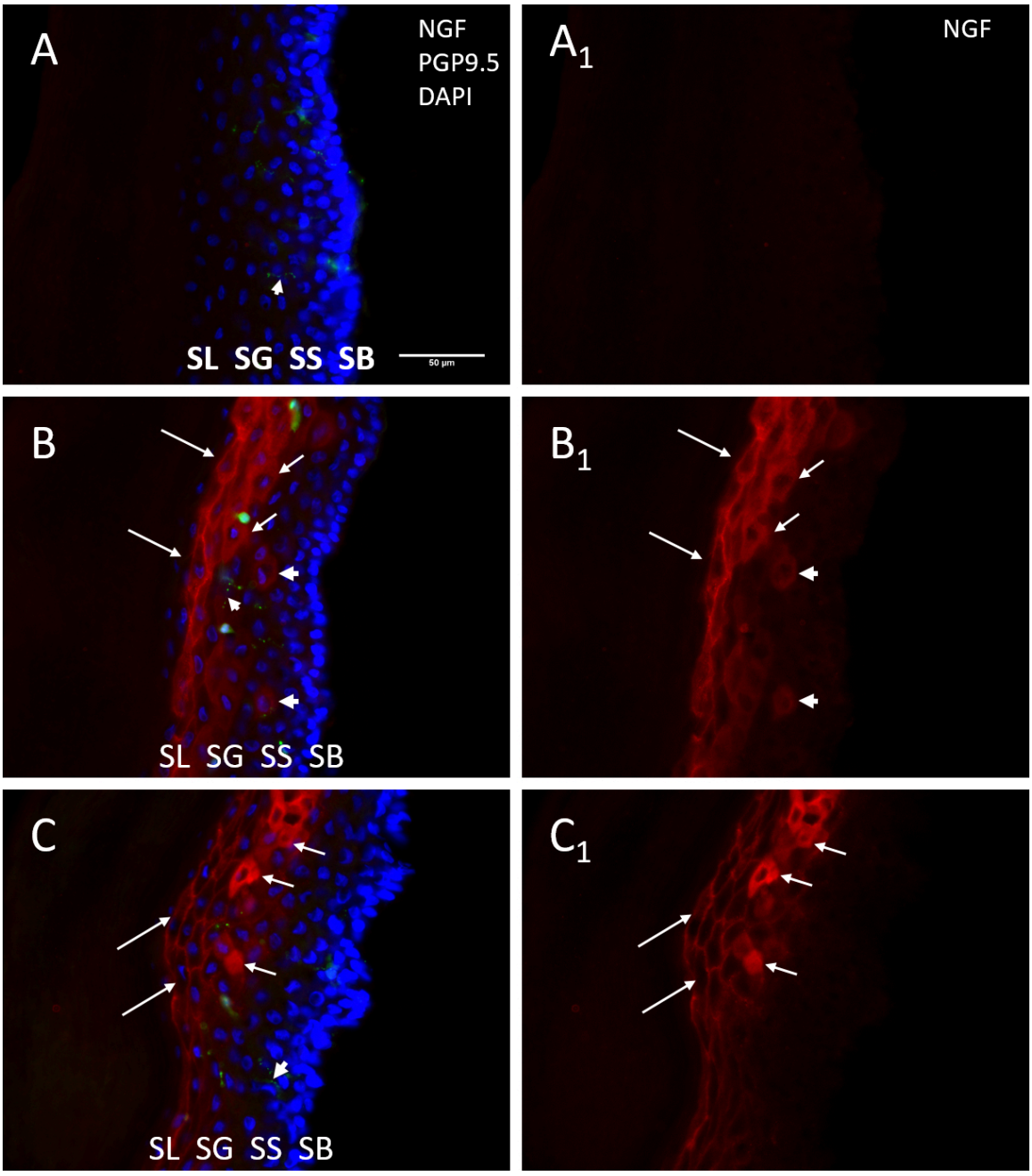


Figure 4.5 NGF and PGP9.5 immunoreactivity. NGF and PGP9.5 immunoreactivity (ir) during **C-II**. Columns **A,B** and **C** show NGF-ir, PGP9.5-ir, and DAPI nuclear staining, whereas columns **A1-C1** show only NGF-ir. In all images, stratum corneum is towards the left and stratum basale towards the right. NGF-ir was not detected in naïve control epidermis (**A**, **A1**). At 6 h **C-II** (**B,B1**), NGF-ir was present in most keratinocytes of the stratum granulosum (short arrows) and

stratum lucidum (long arrows). A few cells for the stratum spinosum were NGF-ir at 6 h **C-II** (large arrowheads). At 12 h C-II (**C,C1**), NGF-ir occurred in keratinocytes of the stratum granulosum and stratum lucidum (long arrows) with some cells in stratum granulosum intensely NGF-ir (short arrows). At no timepoint was NGF-ir detected in stratum basale or stratum corneum. PGP9.5-ir intraepidermal nerve fibers were present in the separated epidermis from naïve and C-II rats (small arrowheads, **A-C**). SB: stratum basale, SS: stratum spinosum, SG: stratum granulosum, SL: stratum lucidum. Scale bar = 50 μm.

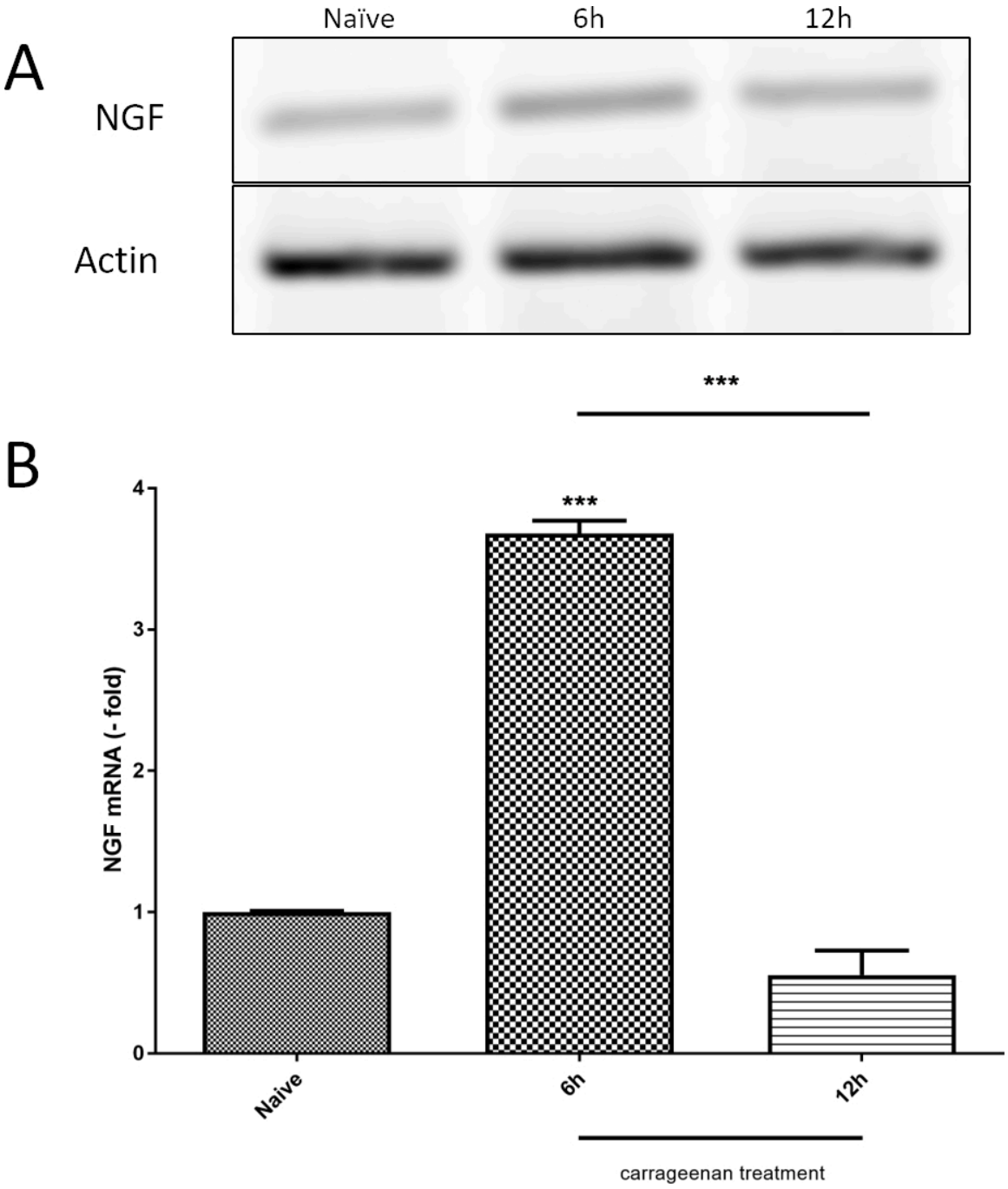


Figure 4.6 NGF mRNA expression during carrageenan. NGF mRNA expression in thermolysin-separated epidermis during C-II was evaluated by qualitative PCR (**A**) and quantitative real-time PCR (**B**). Qualitative mRNA blots (**A**) for NGF and actin demonstrated that

there was good quality mRNA that could be evaluated during inflammation. NGF mRNA expression in epidermis during C-II was evaluated by quantitative real time PCR using actin as a housekeeping gene (**B**). NGF mRNA was significantly elevated (>3-fold) after 6 h of C-II compared to naïve untreated rats, but levels returned to baseline at 12 h (**B**). Results are expressed as the SEM with three rats per group (*$p < 0.05$, **$p < 0.01$, ***$p < 0.001$; student's t-test, unpaired, two-tail was performed at each timepoint).

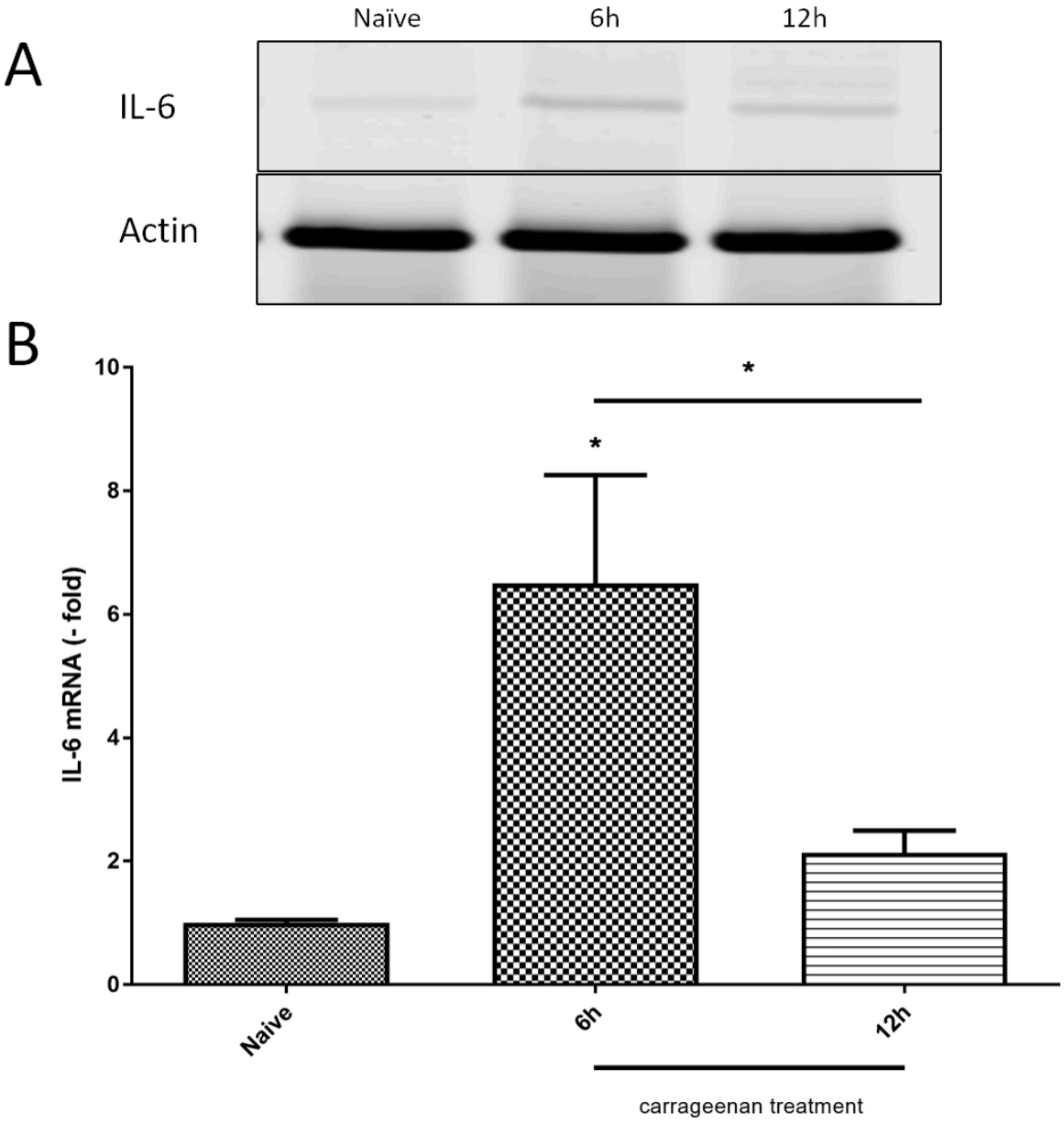


Figure 4.7 IL-6 mRNA during carreneenan. IL-6 mRNA expression in thermolysin-separated epidermis during C-II was evaluated by qualitative PCR (**A**) and quantitative real-time PCR (**B**). Qualitative mRNA blots (**A**) for IL-6 and actin showed there was good quality mRNA for evaluation during inflammation. IL-6 mRNA expression in epidermis during C-II was evaluated by quantitative real-time PCR using actin as a housekeeping gene (**B**). IL-6 mRNA was

significantly elevated (>6-fold) after 6 h of C-II compared to naïve untreated rats (**B**). At 12 h, IL-6 mRNA levels were reduced significantly from 6 h levels but remained elevated (2-fold) compared to naïve rats (**B**). Results are expressed as the mean S.E.M. with two rats per group (*$p < 0.05$, **$p < 0.01$, ***$p < 0.001$, student's t-test, unpaired, two-tail was performed at each timepoint).

CHAPTER V

Three-dimensional reconstruction and quantification of PGP9.5-immunoreactive, sensory intra-epidermal nerve fibers

Intra-epidermal nerve fibers (IENFs) are the peripheral component of primary sensory neurons and are the anatomical attributes for an organism's ability to sense the environment, i.e., tactile, temperature stimuli. These small, unmyelinated, IENFs also transmit protective noxious information critically important to the survival of an organism. Decrease of IENFs within peripheral tissue may indicate potential disease. Several life-threatening peripheral neuropathies, such as autoimmune diseases and diabetes, can be assessed with a skin biopsy. Modern skin biopsy techniques involve removing a small portion of skin to evaluate IENF density and branching. It has been reported that IENF swelling, decrease in IENF density, increased branching, and morphological fragmentation of intra-epidermal nerves may indicate potential disease (Anderson and Miller, 2018a).

IENFs branch throughout the epidermis in three-dimensions (3D). Traditional quantitative evaluation of intra-epidermal nerves involves tissue collection, fixation, sectioning, immunohistochemistry (IHC), and tracing/quantification of two-dimensional (2D) images. This technique has been the industry standard for quantification of intra-epidermal nerves in biomedical research and the limitations have largely been accepted.

A novel, efficient, and cost-effective technique for 3D IENF quantification would provide a powerful tool in the study of peripheral nerves. Quantifying 3D objects with 3D measurement provides greater resolution of data per individual nerve fiber. The challenges of 3D IENF imaging and quantification include epidermal isolation, rendering tissue optically transparent for imaging of z-depth, penetration of antibody with fluorescent labels, proper imaging at greater depths (60-180+ μm), and software segmentation and quantification.

Intra-epidermal nerve isolation was achieved by the use of thermolysin from Geobacillus stearothermophilus at 4°C for approximately 2 hours (Walzer et al., 1989). The low temperature of enzyme activity for thermolysin improves protein integrity before it can be preserved in a tissue fixative. Optical transparency of isolated epidermis is achieved through a modified tissue-clearing (TC) technique (Yang et al., 2014). This involves plasticizing the isolated epidermis in acrylamide at a concentration that allows for increased tissue integrity and molecule penetration for IHC fluorescent labeling (Yang et al., 2014). Once IHC steps are complete the resulting tissue is placed in a refractive index matching solution (RIMS), rendering tissue optically transparent and suitable for 3D imaging. All peripheral nerve segmentation and quantification was performed by the free software Image J Fiji (Schindelin et al., 2012).

Optimization in tissue clearing, immunohistochemistry for TC, image acquisition, and 3D segmentation have allowed for full 3D reconstruction and quantification of IENFs from the epidermis of rats. This technique provides the ability to measure 3D IENF swelling volumetrically, surface area, total length, individual branch lengths, number of branches, and mean-grey-intensity or integrated density of multiple co-localized markers from immunoreactive (IR) fluorescence (**Figure 5.1**). This developing technique allows for IENF quantification in 3D and shows the potential to become a powerful tool for researchers and clinicians. The limitation of measuring three-dimensional objects from two-dimensional images has largely been accepted when quantifying the density of intra-epidermal nerve fibers. Through a novel approach that involves enzyme dermal-epidermal separation, specialized processing, tissue clearing,

immunohistochemistry, confocal imaging, and software segmentation I have developed a technique that allows for IENF 3D reconstruction and quantification. Our developing IENF protocol unleashes 2D restraints and provides full 3D visualization with accurate and whole quantification of IENF complexes.

Published August 2018

doi:10.1017/S1431927618007274

Microsc. Microanal. 24 (Suppl 1), 2018

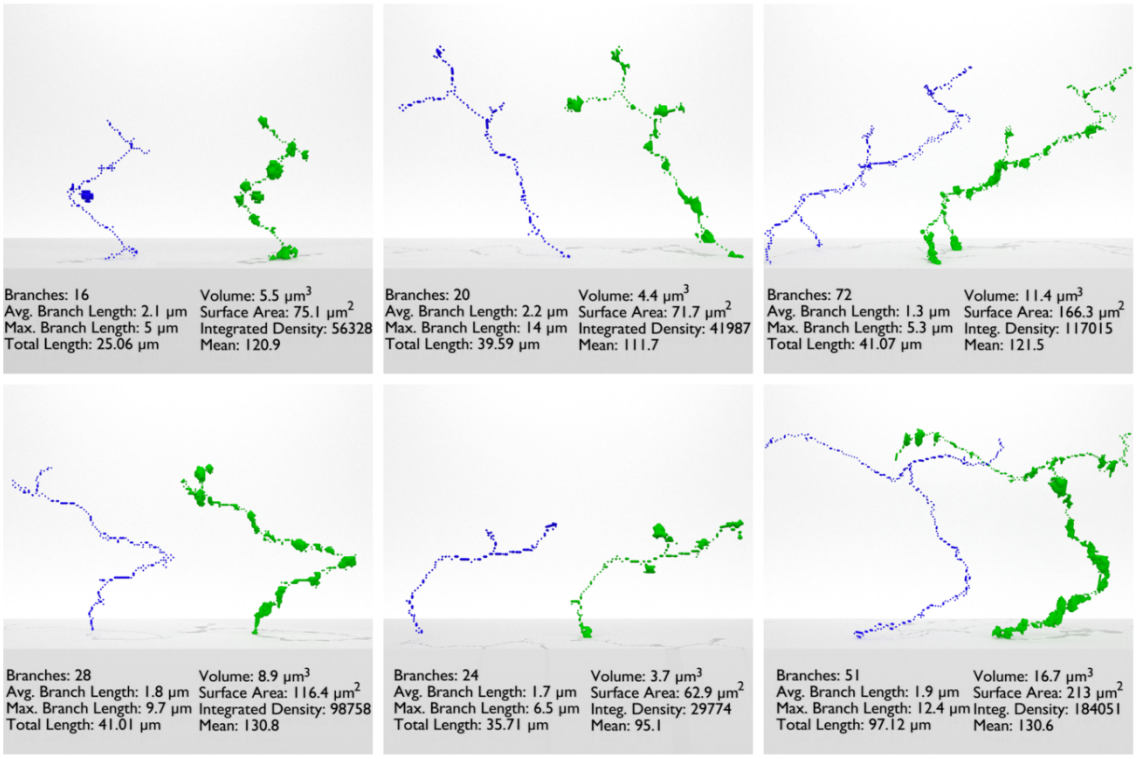


Figure 5.1 Single, isolated, quantified, and 3D reconstructed IENFs. Six examples of individually isolated, 3D reconstructed, and quantified IENFs. Thick green objects are reconstructions based on selective nerve marker protein-gene-product 9.5 (PGP9.5) IR and used for mean-grey-intensity, integrated density, and volumetric measurements. Thin blue objects are skeletonized versions of PGP9.5 reconstructions and used for length and branching measurements. Scale bars calculated per image by dividing hand-traced 2D maximum branch length, in pixels, by software calculated 3D maximum branch length, in microns. The z-depth of traversing nerves (ZN) range from 1.2 – 2.8µm. This provides a scale to approximate 3D to 2D standard conversion.

CHAPTER VI

Passive clarity technique (PACT) optimized for confocal imaging and three-dimensional volumetric measurement and 3D reconstruction of dorsal root ganglion intra-epidermal nerve fibers

Abstract

Epidermal nerve fibers are the peripheral component of primary sensory neurons and allow for an organism's ability to sense the environment, i.e., proprioception, fine touch, temperature and noxious (painful) stimuli. These small, unmyelinated, intra-epidermal nerve fibers (IENFs) provide protective sensory feedback and are critical to the survival of an organism. A normal distribution of IENFs in the epidermis is important for optimally relaying sensory information as well as establishing protective acute inflammation. Reduction in the density of these fibers is an indicator of polyneuropathy and IENFs are clinically evaluated for diagnosis. Traditional evaluation and quantification of IENFs include tissue collection, sectioning, labeling with an antibody reactive to protein gene product 9.5 (a pan neuronal biomarker for IENFs), and measurement of two-dimensional (2D) images. The inherent limitations of measuring three-dimensional (3D) objects in two-dimensions has largely been accepted. However, recent advances in epidermal isolation, optical tissue clearing techniques and immunohistochemistry for cleared whole tissue have allowed

for overcoming previous limitations. I successfully show a novel technique that provides for epidermal and DRG imaging, volumetric measurement, and 3D reconstruction of intra-epidermal nerves and DRG neurons; unleashing antecedent 2D restraints to provide full 3D visualization with accurate and whole quantification of IENF complexes.

Introduction

Skin is composed of the epidermis, dermis, and hypodermis, providing peripheral sensory perception, wound healing, and a water/chemical barrier for all vertebrates. Environmental sensory perception is initiated by axonal projections from neurons in the dorsal root ganglia into all layers of the skin. Wound healing and allodynia (hypersensitivity) are regulated by local interactions between IENFs, keratinocytes, and the immune system, via Langerhans cells (Hsieh et al., 1996; Clayton et al., 2017). The skin's environmental barrier is organized by tightly interconnected epidermal keratinocytes (forming a brick wall-like structure), collagen, and other proteins, providing structure and flexibility. The epidermis has a high degree of both structural integrity and sensitivity. Epidermal nerve fibers for fine touch and pain delicately weave in-between continually differentiating layers of tightly packed and interconnected keratinocytes and are often evaluated in the diagnosis of several diseases (Ebenezer et al., 2007; Pereira et al., 2016; Mangus et al., 2020).

The diverse range of sensory function and intracellular communication elicited by DRG neurons illustrate a highly dynamic and adaptable system which is vulnerable to pathologic alterations. In a disease state, such as diabetes, reduced IENF density can result in a reduction of epithelial tissue neurotrophic factors and a degeneration of epidermal tissue (Timar et al., 2016). Traditionally, whole skin (epidermis/dermal layers) is processed with immunohistochemistry (IHC) and cross-sectioned for two-dimensional (2D) analysis of IENF density, a technique commonly performed by clinicians in the diagnosis of severity peripheral neuropathies (Lauria,

2005). However, the nature of sprawling 3D structures, weaving in and out of a single 2D focal plane, can lead to incomplete IENFs for measurement. IENF fragmentation in 2D images restrict the discrete measurement of total/subpopulations, volume, branching points, and protein content, among other 3D measurements, per nerve. Therefore, a novel technique to evaluate morphologic and IR alterations is sought for volumetric imaging and 3D analysis of IENFs.

Confocal imaging of whole skin for volumetric (3D) measurement can begin from the stratum corneum of the epidermis toward the dermal layers, or from the hypodermis toward the epidermis; however, imaging through the stratum corneum and dermal layers of the skin results in poor IENF imaging quality. To overcome this limitation, separation of dermal layers at the epidermal-dermal papillary junction would allow imaging from the stratum basal of the isolated epidermis toward the stratum corneum. Different methods for separating the epidermal-dermal layers include mechanical, chemical, and enzymatic techniques (Zou and Maibach, 2018). Enzymatic methods can isolate the epidermis from the dermal layers, but these techniques are often biologically harsh and result in degradation of proteins of interest (Jian et al., 2020). To overcome these challenges, I employed a thermophilic-bacterial protease from Geobacillus stearothermophilus to cleave hemi-desmosomes at dermal papillae and isolation of whole epidermis while preserving protein antigenicity (Walzer et al., 1989; Anderson et al., 2021c). The epidermis of the rat hind paw, excluding pads, has five layers of keratinocytes, the stratum basal (composing 4-6 % of all layers), stratum spinosum (16-20 %), stratum granulosum (6-10 %), stratum lucidum (~4-6 %), and stratum corneum (40-61%) (Yousef et al., 2021). The stratum corneum is the thickest layer of the epidermis, providing a water and chemical barrier, often resulting in non-specific fluorescent labeling and photonic obstruction by confocal microscopy (Baroni et al., 2012). Every layer of the epidermis is composed of an interconnected matrix of tightly and spatially bound keratinocytes, adhered by several desmosomes and tight junctions per cell. The stratum corneum and interconnections of deeper epidermal layers provide a robust environmental barrier while also significantly reducing antibody access into superficial regions of

isolated epidermis, such as the stratum granulosum. To overcome photonic obstruction and potentially increase epidermal permeability, the PAssive Clarity Technique (PACT) was optimized for plasticization, removal of lipids (clearing), IHC, and rendering the epidermis optically transparent (Chung and Deisseroth, 2013; Woo et al., 2016; Du et al., 2018) (**Figure 6.1**).

Imaging IENFs from the cleared epidermal stratum basal, superficially, toward the stratum corneum results in a cluster of pixels per individual IENF that collectively move and branch (X and Y plane) through the sequence of 2D confocal slices (Z plane). The fluorescent signal of individual IENFs, represented by a cluster of pixels (a point formed by ~10 pixels), is decreased and fluorescent punctate at the terminal end of an IENF are distant from one another as the IENF extends into the epidermal stratum granulosum. A FIJI script was developed to consolidate protein gene-product 9.5 labeled (PGP 9.5) IENF fluorescent, by enlarging clustered punctate, connecting them in three-dimensions, and deriving the center point of the connected structure, a process of 3D IENF skeletonization. The FIJI macro adds the derived skeleton for each IENF to the original image stack, consolidating each IENF, per field of view, and generating a binary mask of individual and whole IENFs. Generated 3D IENF skeletons are measured for various IENF length and branching measurements. The binary IENF mask is measured for volume, surface area, morphology, and concentration of proteins of interest within IENF mask boundaries, per IENF.

Inflammatory conditions are suffered by millions of people each year, compelling academic and pharmacologic research towards understanding the stages of inflammation. Traditional methods of evaluating inflammation in primary afferent neurons with immunohistochemistry are time expensive, often limiting the scale of the experiment, and can result in untraceable human error. Clinical techniques for the diagnosis of peripheral neuropathic diseases measures the density of intraepidermal nerve fibers and lacks resolution to evaluate patient IENF morphology, branching elements, or protein concentration. In recent decades, computer processors, memory cards, and

graphics cards has experienced tremendous growth, creating opportunity for modernization of traditional scientific techniques.

Methods

Tissue collection

Male and female Sprague Dawley rats (Charles River; n = 8, 150 - 250 g, 8-10 weeks of age) were bred, maintained on-site, and used to evaluate the epidermis with IHC immunolabeling in naïve rats. All rats were given food and water *ad libitum* and housed with a 12-hour on/off light cycle. All methods and techniques utilized in these experiments were conducted in accordance with the National Institute of Health (NIH; https://www.ncbi.nlm.nih.gov/books/NBK43327/), the International Association for the Study of Pain (IASP; https://www.ncbi.nlm.nih.gov/pubmed/6877845) and approved by the Oklahoma State University Center for Health Sciences Institutional Animal Care and Use Committee (2016-03). All efforts were made to care for and minimize the total number of rats used in these experiments to reduce potential suffering.

Epidermal Isolation

The center portion of both rat hind paws (excluding pads) were excised and cut into a 2-millimeter (mm) square with a #11 scalpel blade (Electron Microscopy Sciences, Feather, etc.), stratum corneum facing up. Each square was secured by pinching the stratum corneum with fine forceps and transferred to the well of a 24-well plate, on ice, and floated (dermis down) on 0.1M phosphate buffered saline (PBS). Once collected, all tissue squares were transferred and floated (dermis down) on ice cold thermolysin (500ug thermolysin, 1µg/ml calcium-free tyrodes) and incubated at 4°C for ~2.5 hours, without agitation. One tissue section was transferred at the 2.25-hour timepoint buy securing the dermis (do not secure by stratum corneum) to a petri dish filled

with ice cold PBS. The dermis was secured and submerged below the surface of PBS with forceps (by one hand) and the epidermis was carefully bushed to create an epidermal flap with another pair of forceps (with the other hand). While dermally secured, the epidermal flap was gently pulled by forceps to separate the epidermis from the dermal layers and was placed into a different well with 0.1 M PBS. The epidermis required further incubation if epidermal tearing occurred and less incubation if the epidermis floated away from the dermal region with light touch. Enzymatic activity of thermolysin was deactivated by submerging and incubating the epidermis in Dulbecco's Modified Eagle's Medium (DMEM) with 5 mM ethylenediaminetetraacetic acid (EDTA) for 30 minutes on a rocking shaker (Reliable Scientific, model 55) at 4°C. DMEM solution was aspirated with a 1000 µL pipette (Rainin PR-1000), disposed of, and the well refilled with 4% (w/v) paraformaldehyde (PFA) (0.1 M PBS) and incubated for 1.5 hours on a rocking shaker at 4°C.

Tissue Clearing

Each isolated and fixed epidermal square (2x2 mm) were sliced (corneum down) through the center with a #11 scalpel and the two remaining halves were each cut in the center to yield 4 squares of epidermis, approximately 1x1 mm each. Tissue collected from each rat were placed into individual glass vials (1.85 mL, VWR, 66011-041) and filled (little to no air) with 1% acrylamide in PBS (A1PO) (n=6), or 1% acrylamide with 0.0125% bis-acrylamide (A1PO/bis) (n=2). Air is reduced or eliminated to prevent tissue from sticking to vial/cap and drying out. Glass vials were aspirated and refilled with A1PO six times with a 1000 µL pipette and incubated for 10 minutes each at room temperature on a rocker (Belly Dancer Shaker, Thomas Scientific). After the last wash, glass vials were placed at 4°C and incubated overnight. Micropipettes/tips are preferrable to transfer pipets for all processing steps because tissue is more likely to stick to the porous plastic of transfer pipets can lead to negatively affecting tissue morphology. The next day, glass vials were washed three times with A1PO, for 10 minutes each at room temperature,

and incubated for 2.5 hours in a hybridization oven at 37°C and a rotation speed of nine. A1PO contains a thermal initiator (VA-44) that polymerizes acrylamide monomers at 37°C to generate a soft plastic within the tissue. Tissues were washed 5 times for 2 minutes in PBS and 3 times for 10 minutes in PBS to remove unbound hydrogel solution. SDS solution was placed in the hybridization oven during the warming process to 37°C, before SDS incubation steps. After PBS washes, all tissues were incubated at 37°C in 8% (w/v) sodium dodecyl sulfate (SDS) for 1 hour in a hybridization oven (Thermo Scientific, HS9360) and a rotation speed of nine. To increase the effectiveness of dilapidation (clearing), SDS (37°C) was exchanged in the glass vials, for fresh solution (37°C) at the 30-minute time point. Afterward, tissues were washed 6 times for 10 minutes at room temperature, glass vial lids were discarded and replaced by new lids (to reduce residual SDS), and tissues were washed 6 more times for 10 minutes at room temperature.

Immunohistochemistry

For primary incubation, the antibody was diluted in 0.1M PBS, 0.5% (w/v) bovine serum albumin (BSA), 0.5% (w/v) polyvinylpyrrolidone (PVP), to reduce non-specific binding of antibodies, and 0.3% (v/v) Triton X-100, to dissolve remaining lipids. Polyclonal antibodies, raised in rabbits against protein gene product 9.5 (PGP 9.5), were diluted to 0.5 µg/mL (1:2000) (Ceder Lane Labs, CL7756AP, RRID:) and incubated at 37°C for 48 hours in a hybridization oven and a rotation speed of nine. After primary incubation, tissues were washed 5 times in PBS for 2 minutes each and 5 times in PBS for 10 minutes each, to remove unbound primary antibodies. During secondary incubation, tissues were incubated in 0.3% (v/v) TritonX100 and donkey anti-rabbit Alexa Fluor 488 (Invitrogen, A21206), diluted to 1 µg/mL and incubated at 37°C for 24 hours in a hybridization oven, wrapped in aluminum foil, and a rotation speed of nine. Tissues were washed 5 times in PBS for 2 minutes each and incubated in 4′,6-diamidino-2-phenylindole (DAPI) at a concentration of 300 nanomolar (nM) for 30 minutes at 37°C in a hybridization oven, wrapped in aluminum foil, and a rotation speed of nine. Tissues were wrapped in aluminum foil

in all proceeding incubation steps to prevent fluorophore quenching. Afterward, tissues were incubated 5 times in PBS for 10 minutes each. The lid of each glass vial was removed and positioned upside-down over a glass plate for 5 seconds, until tissue floated down toward the opening of the vial (facing down toward glass plate). Multiple taps on the base of the vial (facing up) resulted in drops of PBS and epidermis falling onto the glass plate. Each epidermal square was secured by the corner of the tissue and carefully placed into the well of a 96-well plate containing refractive index matching solution (RIMS) and incubated at room temperature for one hour on an orbital platform shaker (wrapped in aluminum foil). The refractive index of RIMs solution has a similar refractive index as hydrogel and resulted in near transparency of the epidermis. RIMS was carefully aspirated from each well with a 1000 µL micropipette, refilled with fresh RIMS, and incubated overnight at room temperature on an orbital platform shaker (wrapped in aluminum foil). The next morning, RIMS was aspirated from wells and replaced with fresh RIMS and placed on a rocker (with aluminum foil) while preparing capillary slides (Fisher Scientific, 15-188-51). A small volume (~300 µL) of RIMS was placed onto the glass plate and smaller volume (~100 µL) of RIMS was placed on the capillary slide. Each epidermis was carefully removed from the well and placed into the pool of RIMS solution on the glass plate. The corners of the stratum corneum may slightly curl toward to stratum basal, indicating the orientation of the epidermis. The stratum basal is required to face toward the coverslip for imaging; however, if there is a lack of corner curling, the tissue can be cut through the middle and each piece can be placed on the capillary slide on opposite sides, to assure that one of the two pieces have correct orientation for imaging. Capillary slides were covered slipped (VWR, 82027-788) and placed at room temperature in total dark for 48 hours, or until slides are fully cured and ready for imaging.

Confocal imaging

8-bit image stacks were collected on a Leica SPE TCS confocal microscope (Scanhead RYBV) with a 40x objective (get objective info, oil) at a pixel resolution of 1024x1024 and saved in *.tiff Z-stack (multipage) format. Camera exposures were determined before data collection and were unchanged throughout the imaging process. Z -step slice increments were set to optimal default values in Leica software. Fields of view were manually scanned for areas populated by several IENFs and Z-depth was determined based on PGP 9.5 IR (laser intensity, gain). DAPI (laser intensity, gain) was imaged to visualize keratinocyte nuclei. Collected images were exported as TIF image stacks for FIJI processing.

Image Processing with FIJI scripts

Three IENF scripts were developed on a Windows operating system in the FIJI macro language to evaluate pixel intensity in image stacks, provide global dimming for image stacks with excess pixel intensity, and volumetrically measurement of individual IENFs. A **stack evaluation script** was developed to measure and export the mean of each image slice in a stack to a database file (*.csv) for identifying slices containing excessive background immunofluorescence, which may lead to error. Image slices with excess pixel intensity are manually removed with the slice remover function in ImageJ and saved with the word "_qualified" at the end of the image stack label, for identification. Additionally, the mean of each stack was manually calculated to identify stacks with excessive pixel intensity. Image Stacks with excessive pixel intensity are processed once through the **dimmer script** and saved with the word "_dimmed_qualified" at the end of the label. The dimmer script adjusts brightness/contrast units (20, 275) for reduction of excessive pixel intensity.

The **IENF resolver script** processes qualified image stacks for single IENF isolation, measurement, and exportation of 3D geometry for high resolution qualitative analysis. The

labeling pattern of PGP 9.5 IR from a single IENF is visually characteristic of beads (punctate) on a string (fiber between punctate). In the stratum basal and stratum spinosum, PGP 9.5 labeled IENF punctate/fibers yield strong IR emission (visual contrast), however, in the stratum granulosum, PGP 9.5 labeled IENF punctate yield strong IR emission and fibers have weak (to nonexistent) IR emission. To overcome this limitation, an IENF resolving script was developed to remove background noise and expand the pixel footprint of PGP 9.5 IR, by joining the punctate in the superficial layers of the granulosum. The entire structure was skeletonized, measured for length/branching, and combined with the original PGP 9.5 image stack, consolidating the entire structure of individual IENFs. The resulting hybrid stack was measured volumetrically with the 3D Object Counter (https://imagej.net/plugins/3d-objects-counter). A folder hierarchy was constructed for algorithmic processing and results were exported as .csv files and automatically saved into the data folder. Volumetric measurements were collected directly from 3D Object Counter and high-quality 3D resampled geometry was exported in wavefront (*.obj) format with a resample value of 1. The volumetric mean voxel intensity was calculated from the CGRP channel, based on the mask ROI generated from PGP9.5-IR. A CGRP threshold was determined to classify IENFs as CGRP+ (positive) and CGRP- (negative.

IENF Scoring

The IENF resolving script exported a version of every stack for scoring accuracy. Upon operation, a threshold version of the original image stack (visualized as red) is combined and saved with the image stack of algorithmically identified IENFs (visualized as yellow).

IENF Size Populations

Measurements for volume, surface area, and branching number were collected from segmented 3D objects and organized into 5 groups: complete IENFs, partial IENFs, fragments, clumps, and Langerhans cells. The frequency of IENF distribution was evaluated for each group to determine

size population parameters for collected 3D objects. Proportions based on size for each group were calculated to determine a control profile for PGP9.5-IR in rat epidermis.

Blender Processing

PGP 9.5 IR-generated wavefront (.obj) files were imported into Blender 3D rendering software 2.92 (open-source and free) for visualization and processing (https://www.blender.org). The imported IENF geometry was inspected with the "face orientation" function and geometric normals were flipped to the correct orientation for proper assignment of materials. All vertices were smoothed three times to optimize visualization. Whole IENFs (with identification #) were selected in Blender's edit mode with the "L" key and separated with the "P" key. The origin of individually separated IENFs (with identification #s) were centered to the 3D cursor by moving the origin of the geometry to the 3D cursor and then moving the geometry to the origin point. For proper assignment of materials and lighting, the geometry of IENFs were unwrapped, using SmartUV, and a lightmap pack was generated. Lighting, cameras, materials, and other parameters were carefully set for high-quality rendering in blender; otherwise, the materials were removed for use in the Unreal Engine. Volumetric IENF measurements were collected directly from 3D Object Counter and resampled geometry were generated to illustrate IENF 3D Blender rendering and showcase the IENF Virtual Tour software. Processed IENFs for Unreal Engine were exported as *.FXP (limited to selected objects).

Unreal Engine: IENF Museum of Living Art

Unreal Engine is a free and powerful software capable of photorealistic rendering, animation, physics, and artificial intelligence, **in real-time** (https://www.unrealengine.com/). Unreal Engine is currently used as a powerful tool in the development of next-generation console and personal computer games, films, visualizations, physic simulations, and in architecture.

Museum architecture was purchased from SCL in Unreal Marketplace for Unreal Engine 4.27 and served as a foundation to customize and build upon (https://unrealengine.com/marketplace/en-US/product/scl-assets-04-museum?lang=en-US). The museum was scaled up to create a desired space. Museum geometry was imported into Blender software to customized stairs, pool railing, and to lower the depth of the lobby fountain. Tables and pedestals were designed in Blender and models were imported into Unreal Engine for material application and placement.

Results

Epidermal Isolation

Enzymatic isolation of whole epidermis with thermolysin was achieved at the 2.25-hour timepoint and cross-sectionally evaluated for accuracy by staining with toluidine blue. Most of the epidermis separated well but a few pieces, most were near to pads, did not separate efficiently. Only efficiently separated isolated epidermis were retained for tissue clearing and IHC.

Tissue Clearing

Several isolated epidermal pieces floated before polymerization and sank afterwards, increasing the ease of PBS washing and changing solutions during antibody incubations. At the thirty-minute and one-hour time point, the SDS solution contained small pieces of white material which slowly sank to the bottom of the glass vials. To assure correct orientation for imaging transparent epidermis, each 2x2 millimeter tissue section was sliced in the middle, one piece was flipped directly before cover slipping.

Epidermal Imaging of PGP9.5 and CGRP-IR

Intraepidermal nerve fibers and Langerhans cells were immunoreactive for PGP9.5 antibodies. CGRP was immunoreactive only within a subpopulation of PGP9.5-IR IENFs. The stratum basal (deepest layer) contained more PGP9.5_IR non-specific background than superficial layers. There was low to non-existent levels of background for CGRP-IR in all levels of the epidermis.

ImageJ/FIJI Scripting for acquired Fields of View

Acquired fields of view were collected over a time span of two years. The IENF resolver script was developed after image collection and the primary variation within all image stacks were differences in pixel intensity. The IENF resolver script processes images based on pixel intensity, therefore these variations resulted in a loss of complete and partial IENFs. To overcome these limitations, a stack evaluation script calculated and exported the mean of each slice (within image set/field of view) and the total mean from each slice was manually calculated. It was determined that slices with a mean of 12 or above are *exceedingly bright* and marked for removal with the slice remover function in FIJI. Image stacks with a total averaged mean of 10 or above were identified as *exceedingly bright* and saved with the string "_dimmed" added to the end of the label in the *.tif file. Image stacks containing slices with a mean below 12 were saved with the string "_qualified" added to the end of the label in the *.tif file. Image stacks containing "_dimmed" in the title were processed by the dimmer script to reduce the overall pixel intensity into a range optimal for the IENF resolver script. Image stacks were processed a total of once and saved with the script "_dimmed_qualified" added to the end of the label in the *.tif file. All "qualified" image stacks were processed by the IENF resolver script.

IENF Scoring

Merging binary channels for noise (red) and PGP9.5-IR (yellow) a visually distinct and contrasting image stack was generated and visualized in the 3D Viewer categorizing and scoring all identified objects (**Figure 6.2**). PGP9.5-IR objects were categorized as complete IENFs,

partial IENFs (~90% complete), fragment, or clump. Partial IENFs contain ~90% yellow (measured portion of IENF) and small extensions in red (unmeasured portion of IENF), providing a distinction between complete and partial IENFs.

Out of 453 PGP9.5-IR identified objects, 169 were complete (37.31%), 98 were partial (21.63%), 124 were fragments (27.37%), 23 were clumps (5.08 %), and 39 were Langerhans cells (8.61%). Collectively, complete/partial IENFs (combined) constituted 267 (58.94%) of the total PGP9.5-IR objects identified.

Frequency Distribution of IENFs

Small, medium, and large size populations for PGP9.5-IR objects were based on data collected from complete, partial, and complete/partial IENFs. Frequency distribution for volume, surface area, and branching number showed a similar histogram pattern, containing a tall peak and small elbow. Small and medium size populations were divided between the ascending side (small) and descending side (medium) of the tall peak. The large population was everything to the right of the medium population and included the small elbow. For volume, the population of small ranged from 0 – 200 μm^3, medium ranged from 201 – 400 μm^3, and large ranged from 401+ μm^3 (**Figure 6.3**). For surface area, the population of small ranged from 0 – 1000 μm^2, medium ranged from 1001 – 2000 μm^2, and large ranged from 2001+ μm^2 (**Figure 6.4**). For branching number, the population for small ranged from 0 – 50, medium ranged from 51 to 100, and large ranged from 101+ (**Figure 6.5**). There were differences in CGRP+ populations between volume, surface area, and branching. CGRP populations were spread among small, medium, and large populations for volume, surface area, and number of branching points. Fragments and clumps were primarily in the small and medium populations, whereas Langerhans cells were more dispersed between all sizes, with a preference for small and medium for branching populations.

CGRP-positive IENFs

The mean of CGRP-positive and CGRP-negative populations were consistent between complete, partial, and complete combined with partial groups. The group mean for complete CGRP-negative and CGRP-positive IENFs were 14.3 and 83.7, partial IENFs were 12.5 and 82.4, and complete combined with partial IENFs were 13.4 and 83.2, respectively (**Figure 6.6**).

Unreal Engine: IENF Museum of Living Art

The museum was designed to showcase each category of PGP9.5-IR objects as exploratory exhibits. Image stacks were identified in the "IENF Scoring" folder and good examples of each group were selected for reconstruction and display. Selected image stacks were imported into FIJI, split into separate channels, and saved as a "noise" channel (exported as red) and "IENF" channel (exported as yellow). The geometric surfaces of all reconstructions were exported with a resample rate of 1 (highest geometric resolution in 3D viewer). This process was achieved to clearly differentiate noise from IENFs and illustrate how PGP9.5-IR objects were scored (**Figure 6.7**).

Discussion

The first documented compound microscope was invented by Antony Van Leeuwenhoek in the mid-seventeenth century and was quiet literally a lens into a micro-universe previously out of observation (Singer, 1914). This powerful tool was immediately useful to biomedical 16th Century scientists, inspiring other technologies. Nearly 100 years later, in 1770, George Adams, Jr. ushered a new age of histology by inventing the first, hand operated, microtome (cutting device). Since then, tissue sections of varying thickness have been used to evaluate cross-sectional tissue morphology and proteins of interest in histologic samples. In the 20th Century, two papers were published describing the first confocal laser point scanning microscope and

opened the doors the volumetric 3D imaging. However, tissue opacity limits excitation light traveling toward antibody-bound fluorophores (within tissue) and light emission traveling back toward the microscope detector, limiting imaging depth. As in the past, the invention of the confocal laser scanning microscope further inspired other technologies (Cabral and Moratti, 2011; Chai et al., 2017; Gradinaru et al., 2018). Early tissue clearing research by a 20th Century scientist, Werner Spalteholz, juxtaposed by a need for transparent tissue, for confocal microscopy, led to the development of a method of making fixed tissue samples transparent using acrylamide-based hydrogels (CLARITY) (Chung and Deisseroth, 2013; Du et al., 2018). These discoveries, combined with open-source algorithmic image analysis, provide a reliable set of tools, today, capable of imaging discrete volumetric and voxel-based measurements in fixed epidermis.

The DRG contains sensory neurons, projecting axonal processes from the cell body (near the spinal cord) to innervate the epidermis in complex sensory networks. The stimuli-driven functional domain of IENFs are responsible for an organism's ability to protect itself from tissue injury and to better interact with the environment. Distinct types of IENFs have varying thresholds of depolarization (action potential signaling), however, previous methods of IENF measurement were limited to collecting data on cross-sectional density, lacking overall resolution to discriminate between IENF types (Woolf and Ma, 2007). Sample preparation for cross-sectional (2D) IENF image analysis (traditional method) takes less than a day and accurately provides densitometric measurements, based on area. Morphologically, intra-epidermal nerve fibers belong to a network of branching 3D structures, intricately weaving throughout ever-differentiating beds of epidermal keratinocytes and require volumetric (3D) image analysis to collect accurate data points, per IENF.

Thermolysin enzymatic separation of rat hind paw epidermis was successful and samples with suboptimal separation were discarded (Anderson et al., 2021c). Methods for tissue clearing and

immunolabeling were modified from the CLARITY technique (Chung and Deisseroth, 2013; Du et al., 2018). The electrophoretic field for increasing the speed of delipidation during the clearing step was omitted to preserve the integrity of the epidermal stratum basal. A lower concentration of acrylamide (1%), compared to 4% in the CLARITY technique, with 0.0125% bis-acrylamide was determined to be optimal for thermolysin-separated epidermis. I found that 1% acrylamide (A1PO) produced deeper and more contrasted fluorophore immunoreactivity than 4% acrylamide (A4PO) in the epidermis. At the 30-minute and final time point of the clearing step, with 8% SDS, the solution in the glass vials presented with a slight cloudy appearance, as a result of delipidation.

Capturing images from the stratum basal toward the stratum corneum is optimal when imaging the epidermis with a confocal microscope. The isolated, cleared, and immunolabeled epidermis was fully transparent, making it difficult to determine the proper side for imaging. Before cover-slipping, samples were transported to a small pool of RIM solution on a glass plate, to keep samples hydrated, and nudged out into the air on a glass slab. Over the course of the experiments, based on light refraction, corners of some square samples appeared to have a slightly curl. The corners of the rigid stratum corneum may be curling toward the less rigid lower layers, however, to assure images were available for each processed sample, every epidermis was sliced in half and one half was flipped over, before cover-slipping.

The combination of PGP9.5 and secondary (AlexaFluor 488) antibodies produced non-specific immunofluorescence in the stratum basal, when compared to layers in the stratum spinosum to stratum granulosum. However, the combination of CGRP and secondary (AlexaFluor 555) antibodies used in these experiments produced low non-specific background in all layers of the epidermis. A system was developed to identify confocal image slices containing excessive non-specific PGP9.5-IR labeling. A script was developed to evaluate the mean of each slice from image stacks to identify slices for manual deletions, of slices with a mean of 12 or above. The

mean of the entire stack is manually determined and image stacks with a mean of 10 or above and processed once through a dimmer script and all "qualified" imaged stacks are processed by the IENF resolver script. The resolver script generates 11 folders of processed images and 1 data folder, with a *.csv file of collected data. The concept of the IENF Scoring folder was to produce a version of the field of view (image stack) suitable for discriminating PGP9.5-IR objects. A merge of the *noise* channel (red), based on the original image stack threshold at 25 histogram units (8-bit image, 0-255), and the *IENF* channel (yellow), based on measured objects, were automatically generated and exported into the IENF scoring folder. The scored image folder contains a version of the original image stack and can be used to discriminate complete IENFs, partial IENFs, fragments, clumps, and Langerhans cells (and their processes). The overall accuracy for automatic identification of IENFs was low (37.3%), however, the IENF image stacks for scoring allowed for quick scoring and accurate data curation.

The frequency distribution of volume, surface area, and number of branch points were evaluated for complete IENFs. A similar pattern emerged from each group of a tall peak and a small elbow. Size populations were established for small, medium, and large populations based on the tall peak and elbow for volume, surface area, and number of branch points. CGRP-positive IENFs were in all size populations for volume, surface area, and number of branching points. Several factors (IR-based artifacts) can artificially increase or decrease the number of branch points during the skeletonization process, therefore, volume and surface area may be more appropriate to measure IENFs with the resolver script. Additionally, the mean difference of CGRP-positive and CGRP-negative IENF fibers between groups for complete IENFs and partial IENFs was 1.8 histogram units (8-bit image, 0-255), indicating complete and partial IENF data have similar resolution.

To visualize and study individual IENFs, a 3D environment was generated, in the form a digital museum. The IENF museum of Living Arts was developed in Blender (2.93.4) and Unreal Engine 4.27. Image stacks were imported into Blender, scaled by 0.01, materials removed,

normal flipped, and exported as an FX Bank file (*.fxb file format). Character models were also imported from the Adobe Maximo and used to generate scenes and create ambience. Accurate scale bars were imported and placed around the museum for reference of scale. Several exhibits comparing the size of red blood cells and the thickness of human hair to the scale bars provided reference of scale. IENF models were based on images generated by the IENF resolution script, (IENF scoring folder) and imported as two separate 3D meshes, a red *noise* channel and a green *IENF* channel. Each IENF exhibit is slowly rotating and the information for that nerve is descripted on the pedestal below. The IENF museum of Living Arts was designed to be highly educational and possibly a vision into the future of VR science conferences. Unreal Engine is currently used as a powerful tool in the development of next-generation console and personal computer games, films, visualizations, physic simulations, and in architecture. This research adds another dimension for the use for the Unreal Engine, biomedical research.

IENFs are the stimuli-driven functional domain of 1st order, primary afferent sensory, DRG neurons. Standardized and accurate IENF volumetric measurements, combined with respective 3D reconstructions, were the goal of this research. Intricacies involved in the calcium-influx-driven depolarization and generation of action potentials, traveling towards third-order neurons responsible, from the perception of fine touch to debilitating pain have relevant clinical and research significance. Previous (and currently practiced) methods of IENF measurement provide data for (mostly) IENF density. Methods and results of this research describe a reliable technique to identify and label complete IENFs with 27-37% accuracy, using one filter channel (PGP9.5-IR). Manual scoring for accuracy was observed by visual examination of images-stacks processed by the Masks of Nearby Points ImageJ/FIJI plugin (exported into the IENF Scoring folder), resulting in *noise* 3D meshes (red) and *measured IENF* meshes (green). I believe that a clear identification between complete and partial IENFs is demonstrated in the IENF Museum of Living Arts, a possible view into a future for scientific conference and collaboration.

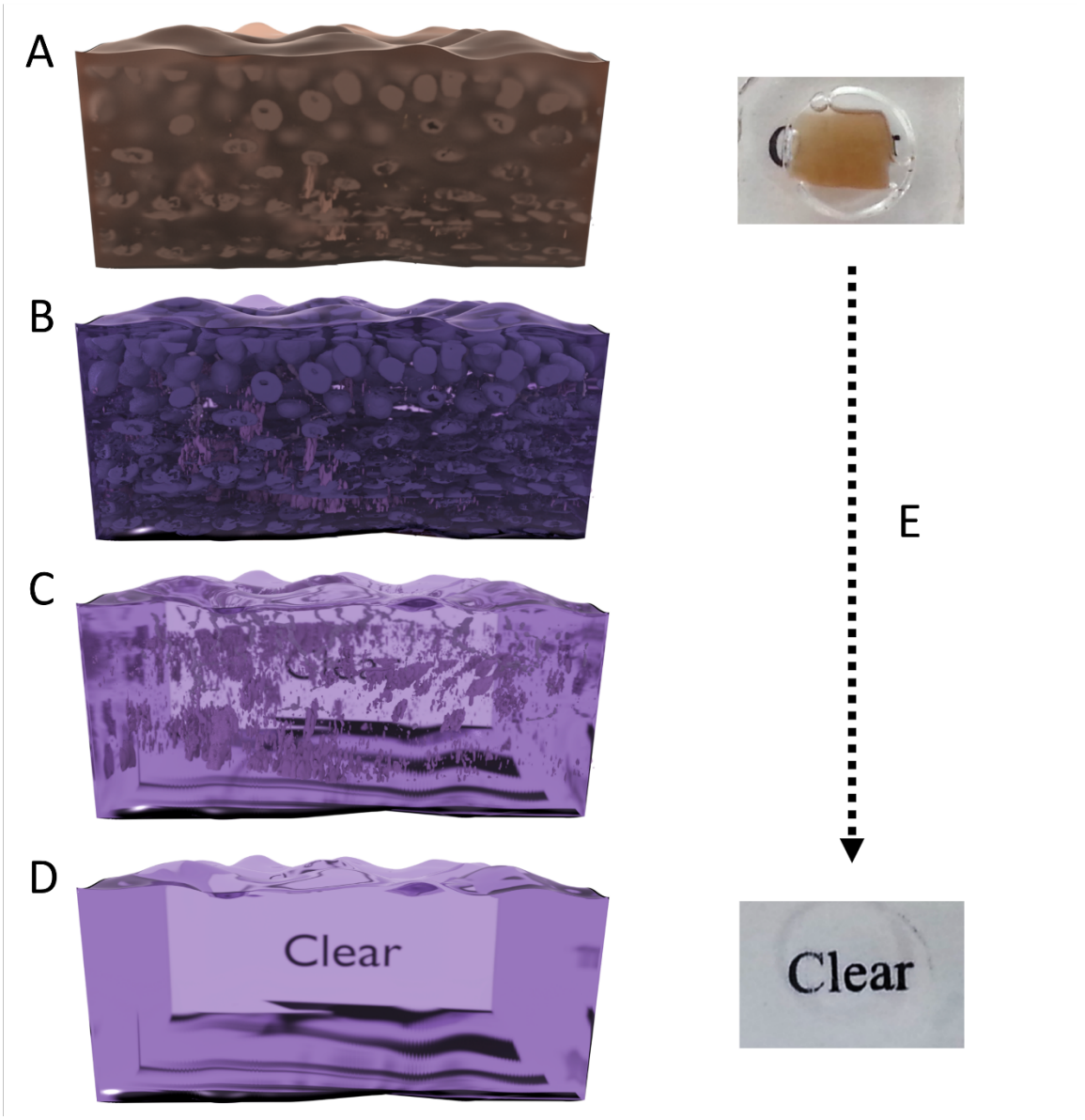


Figure 6.1 Tissue clearing process in epidermis. **A-D**, rendered models; **E**, example of cleared lung tissue. **A**, isolated epidermis; **B**, polymerized and isolated epidermis; **C**, polymerized, cleared, and isolated epidermis; **D**, RIMs polymerized, cleared lung tissue before and after PACT tissue clearing.

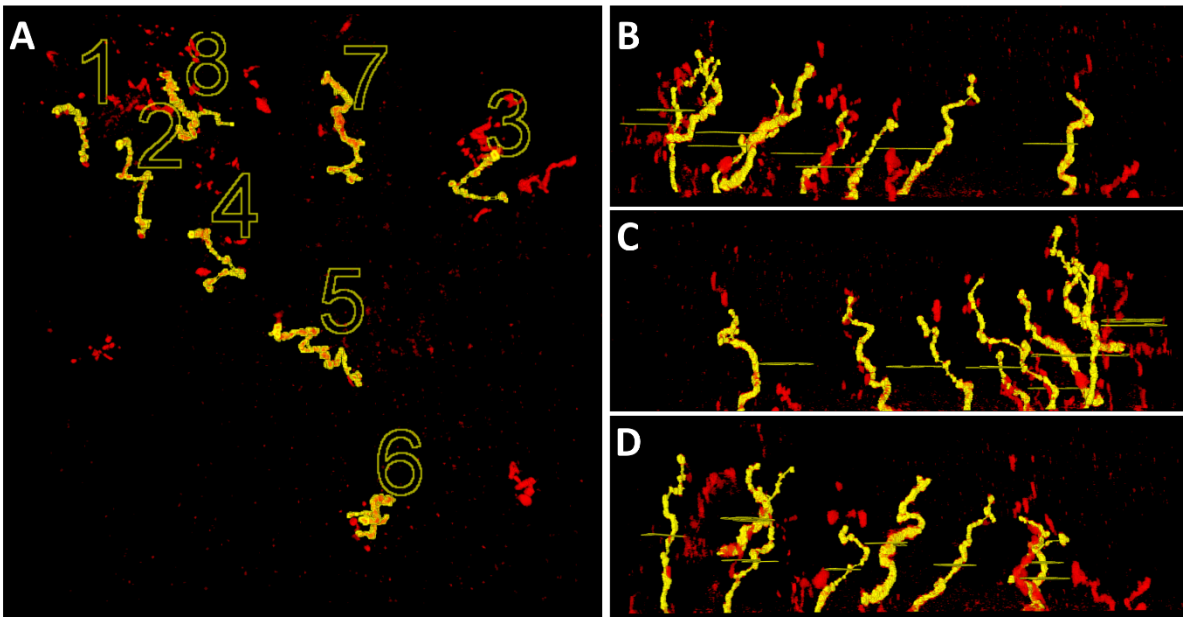


Figure 6.2 Three-dimensional perspective of IENFs. Three-dimensional perspective of intra-epidermal nerve fibers (yellow) merged with overall PGP9.5 IR (red: background and unlabeled portions of nerves). **A**, top-down view of image stack field of view; **B** side view one; **C**, side view two; **D**; side view three. Multiple side views illustrate 3D structure and criteria used for scoring images as complete, partial, fragments, clumps and Langerhans cells.

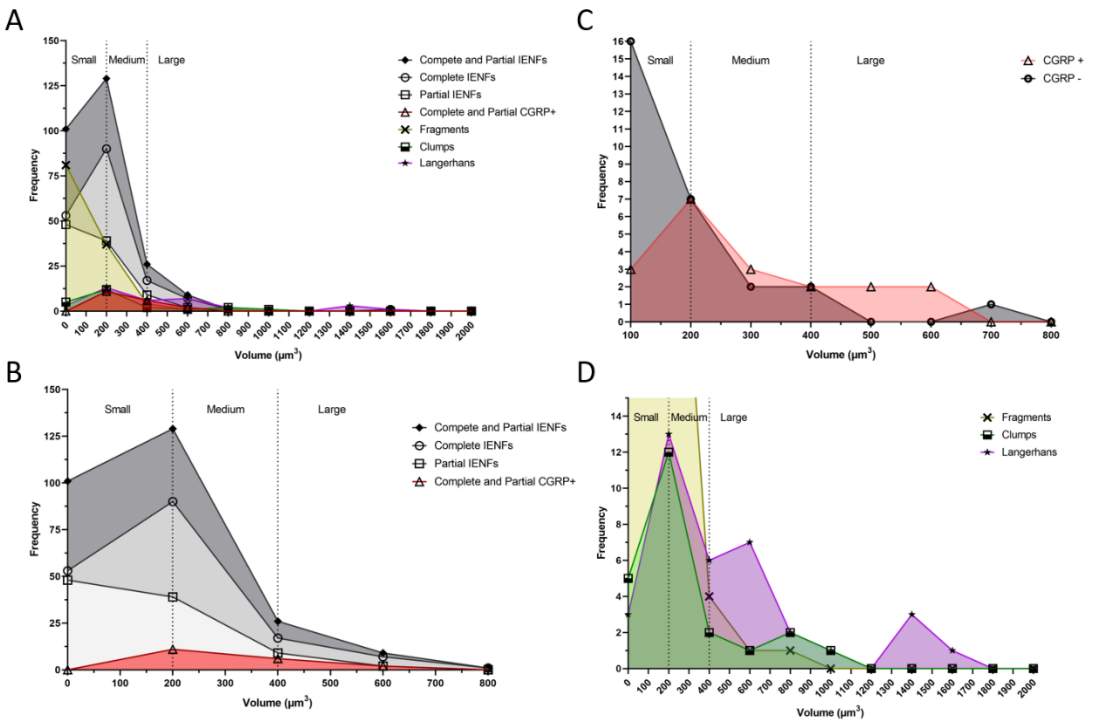


Figure 6.3 IENF size populations, based on <u>volume</u>. A, all categories of PGP9.5 identified objects; **B**, all IENF groups; **C**, CGRP positive and CGRP negative groups; **D**, non-IENF groups.

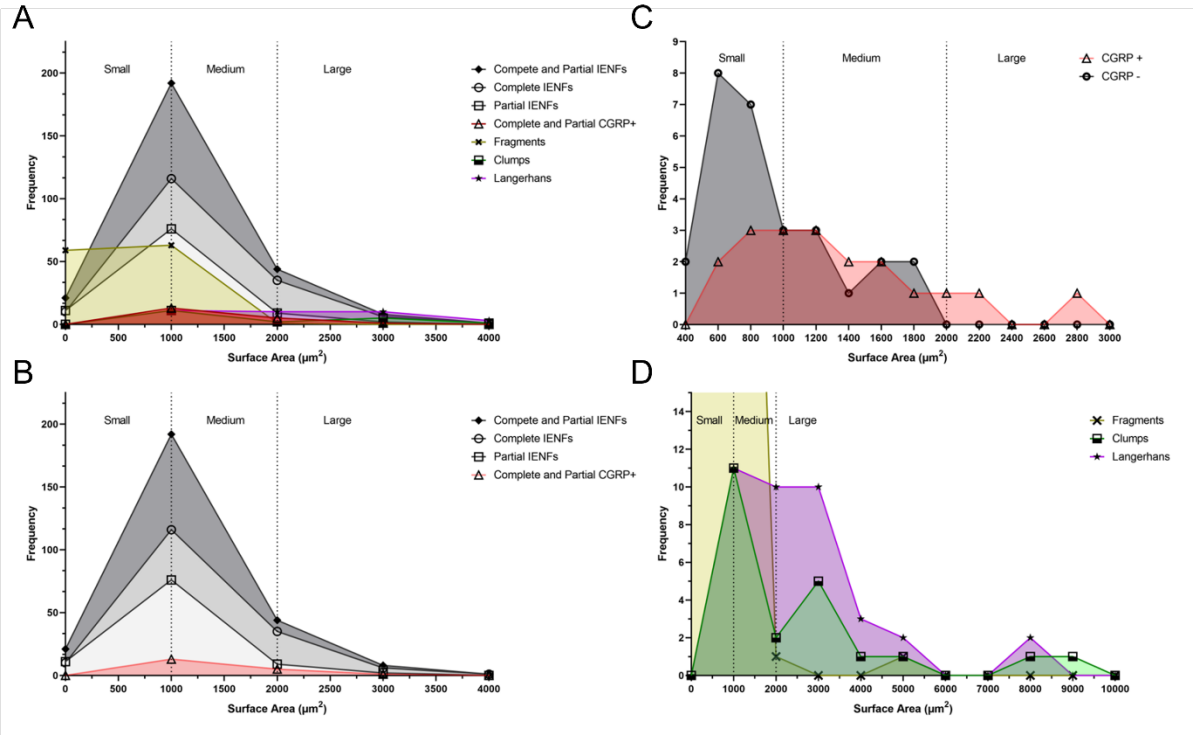


Figure 6.4 IENF size populations, based on surface area. **A**, all categories of PGP9.5 identified objects; **B**, all IENF groups; **C**, CGRP positive and CGRP negative groups; **D**, non-IENF groups.

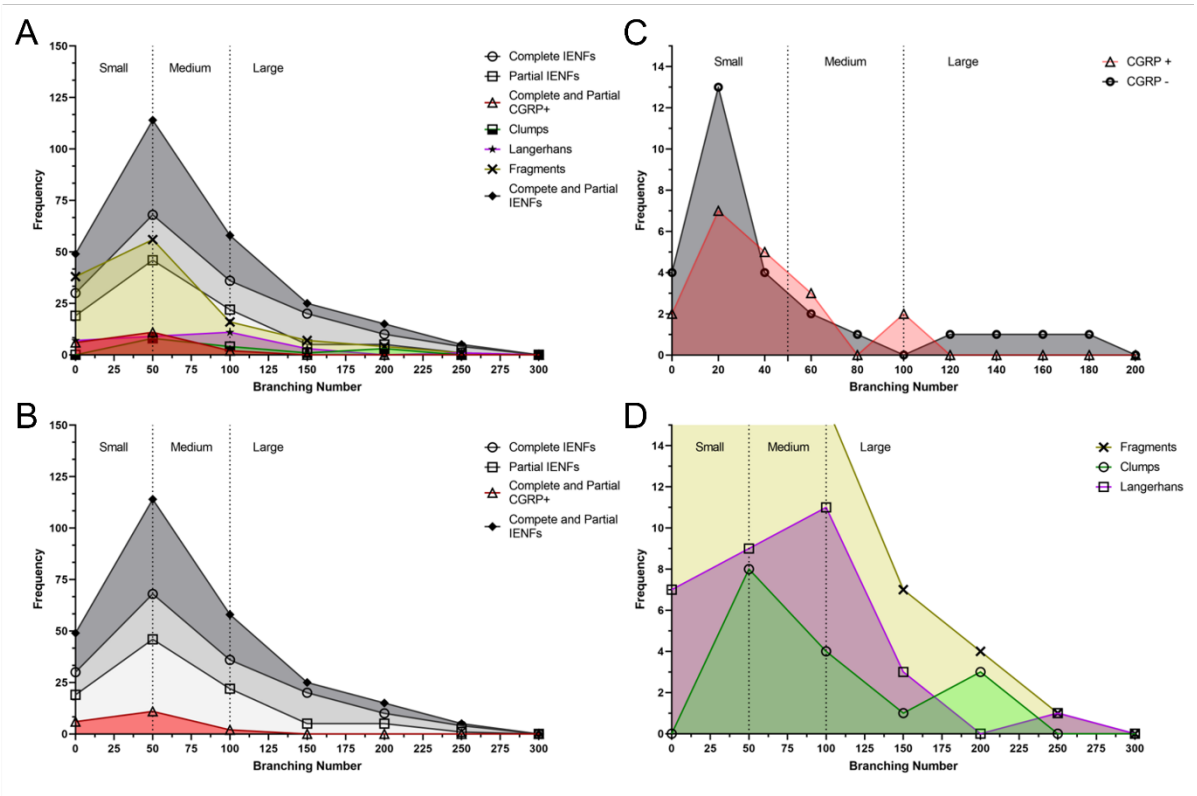


Figure 6.5 IENF size populations, based on <u>total branching number</u>. A, all categories of PGP9.5 identified objects; **B**, all IENF groups; **C**, CGRP positive and CGRP negative groups; **D**, non-IENF groups.

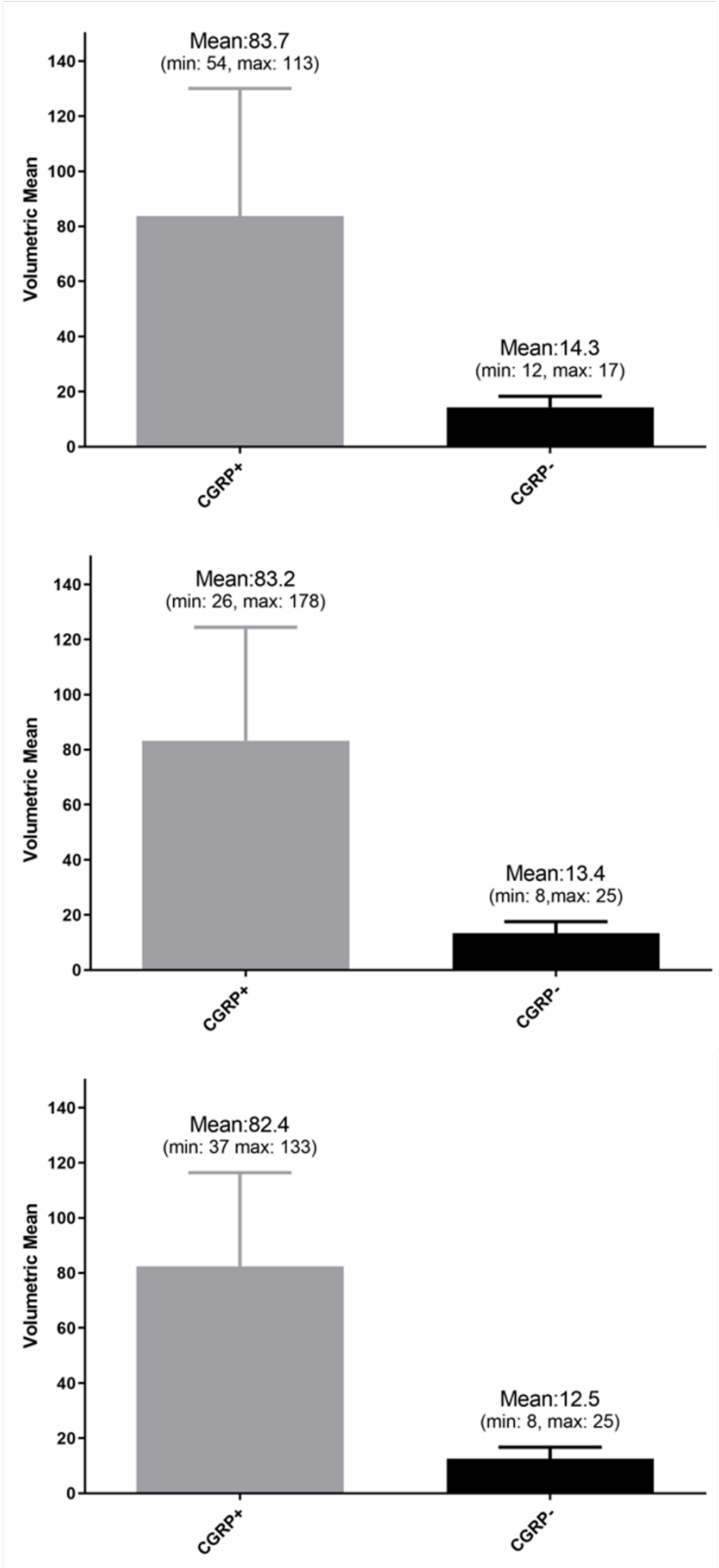
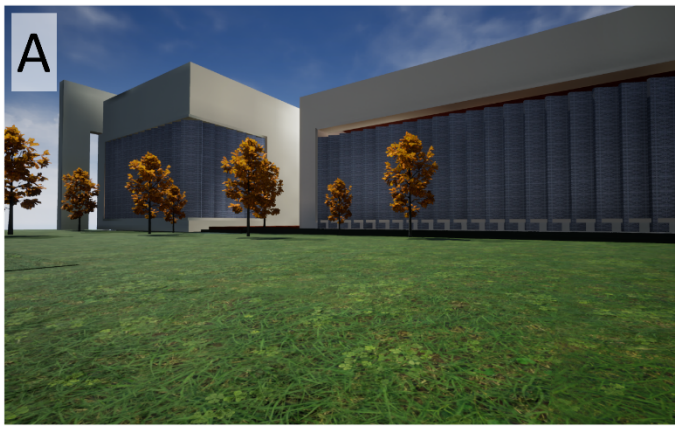
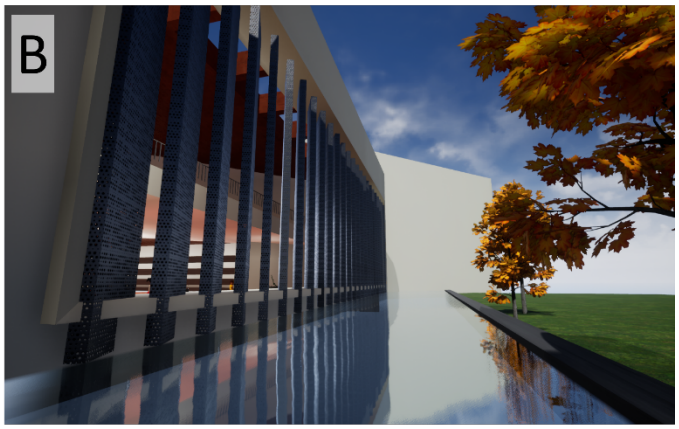

A
Mean:83.7
(min: 54, max: 113)
Volumetric Mean
140
120
100
80
60
40
20
0
Mean:14.3
(min: 12, max: 17)
CGRP+
CGRP-
B
Mean:83.2
(min: 26, max: 178)
Volumetric Mean
140
120
100
80
60
40
20
0
Mean:13.4
(min: 8,max: 25)
CGRP+
CGRP-
C
Mean:82.4
(min: 37 max: 133)
Volumetric Mean
140
120
100
80
60
40
20
0
Mean:12.5
(min: 8, max: 25)
CGRP+
CGRP-

Figure 6.6 CGRP expression of IENFs for volume, surface and branching number.

Volumetric mean of CGRP in CGRP positive and CGRP negative populations for **A**, Complete IENFs; **B**, Partial IENFs; **C**, Complete and Partial IENFs. Minimal volumetric differences in CGRP IR, between complete IENFs, partial IENFs, and complete + partial IENFs, illustrate that data collected from complete IENFs is reflective to complete + partial IENFs.


A


B


C
Did You Know?
70 µm (micrometers) is about the length of
10 red blood cells lined up 1 by 1
Complete IENF
Volume: 96.7 (µm³)
Surface Area: 853 (µm²)
Branching: 269

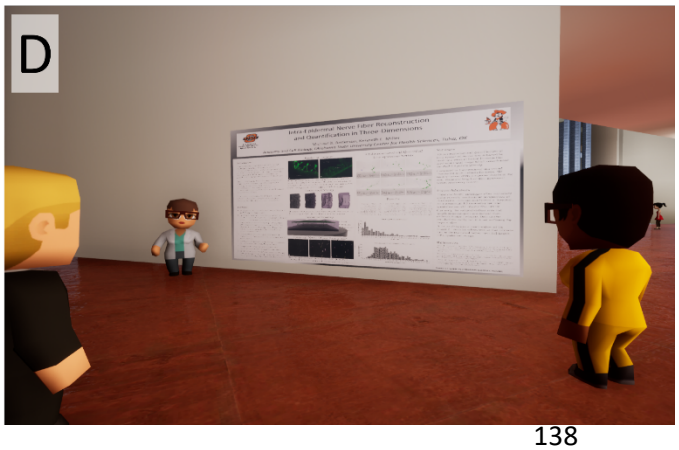

D

Figure 6.7 The Kenneth Eugene Miller Museum of Living Arts. A, outdoor perspective of museum; **B**, outdoor perspective by entrance; **C**, single intra-epidermal nerve with scale bar. The green portion of the nerve is measured, and areas of red were unmeasured, thus complete IENF; **D**, poster presentation of tissue clearing technique (originally presented Society for Neuroscience, 2017). The Kenneth Eugene Miller Museum of Living Arts, main character, and non-playing characters (including animations) were imported in the Autodesk Filmbox format (*.fbx) format. All IENFs (red/green) were imported from ImageJ/FIJI (wavefront format, *.obj), based on a resampling rate of 1, imported into Blender for processing (planar normal flipping), and exported as *.fbx. Fixtures, information boxes, and scale bars were created in Blender and saved as *.fbx. All geometric structures (*.fbx) were imported into Unreal Engine 4.27 and light-refractive materials were assigned to create a semi-realistic, (maybe) intriguing, and educational experience

CHAPTER VII

Conclusion

Chronic pain and inflammation are life-altering symptoms of peripheral neuropathic disease, leading patients towards addictive medication to alleviate (often debilitating) pain. DRG neurons are studied for understanding proteins and pathways involved in chronic cellular homeostatic maladaptation, due to neuropathic disease and injury. Several studies have reported the cautioned use of a biomarker for neuronal apoptosis, NeuN/FOX-3, due to its reappearance after a period of time, in various assault models (McPhail et al., 2004; Unal-Cevik et al., 2004; Collombet et al., 2006; Portiansky et al., 2006; Alekseeva et al., 2015). Ablation of the NeuN-FOX-3 protein, as a result of neuronal assault, and reemergence to normal levels by 28^{th} day signifies neuronal injury and recovery (McPhail et al., 2004). The AIA model in the hind paw of the rat mimics severe peripheral inflammation, persisting for several weeks. Somatosensory DRG neurons project functionally active domains (IENFs) into peripheral tissues, and it was unknown if DRG neurons were apoptotic or were injured. Data collected from DRG tissue with an unobserved reduction in the DRG neurons or neuronal function would be deficient and inaccurate. Neuronal morphology, DAPI-IR, and the NeuN protein were evaluated by immunohistochemistry and protein analysis to determine if the AIA model led to apoptosis or altered neuronal function. At the 48-hour time point of the AIA model, neuronal nuclei and cytoplasmic morphology showed no signs of apoptosis (membrane blebbing), based on DAPI-IR and NeuN-IR, respectively

(Charras, 2008; Andrade et al., 2009; Eidet et al., 2014). Furthermore, NeuN-IR increased from control to AIA conditions in nearly all size and CGRP populations, suggesting a robust neuronal response (increased transcription), to repeated activation of DRG neuronal IENF domains due to inflammation. These results suggest there is no reduction of DRG neuronal integrity at the peak swelling time point of the AIA model; indicating measured neuronal responses to AIA-induced acute inflammation are within range of neuronal function, thus measured protein changes reflect a full response by DRG neurons (Anderson et al., 2021b).

Academic researchers' study DRG neuronal cell bodies in pain and inflammatory research to understand mechanisms involved in neuronal maladaptation after injury, or during disease, and resulting permanent dysfunction leading to chronic pain. Previous methods for quantifying neuronal changes in DRG neurons involved manually hand tracing of DRG nuclear and cytoplasmic boundaries (using a pen-input display monitor or mouse) and takes several days to weeks to complete. Data collected from this technique is accurate but requires excessive time and can be a source of bias. An algorithmic approach to automatically collect data from histologic and IHC-labeled (cryosectioned) tissue sections was unavailable, due to diverse histologic morphology, IHC fluorescent non-specific labeling/artifacts, and a heterogeneous cell population. To overcome these limitations, a series of algorithms were developed to process NeuN/FOX-3-IR labeling, a protein abundantly located in DRG neuronal nuclei and cytoplasm, to qualify DAPI-labeled nuclei as *neuronal nuclei*. The resulting IC-DRG script processes images based on pixel intensity, therefore a method for camera exposure calibration was required to assure accurate use of software. The evaluation mean grey intensity system was developed to assign a value for overall image brightness, serving as an effective method for determining optimal camera exposure time. The resulting IC-DRG script produces accurate (binary) DRG masks for neuronal nuclei and cytoplasm and imported into CellProfiler for measurement with other proteins (channels) of interest, ~24 times faster than the previous technique (Anderson et al., 2021a).

Clinicians use 2D immunohistochemical techniques for measuring IENF density in the diagnosis of peripheral neuropathic disease. Intra-epidermal nerve fibers contain a variety of sensory modalities and are intricately imbedded throughout the epidermis as free 3D branching nerve endings. Measuring the 2D density of IENFs assists in diagnosing peripheral neuropathy, however, volumetric 3D data collection would reveal IENF profiles for healthy and disease states across several measurements, and per IENF. Traditional cross-sectional evaluation of the skin (epidermis, dermis, and hypodermis) and evaluation with IHC did not serve for confocal imaging and segmentation of IENFs in 3D. To isolate the epidermis, an enzyme with a thermal temperature of 4°C was employed to preserve proteins while cleaving hemidesmosomes, adhering the basement membrane of the epidermal stratum basal to the dermis. The isolated epidermis was cross-sectioned and evaluated for IENFs, but the cross-sectional density and limitation of confocal Z-resolution resulted in IENF clumping, and loss of data. It was determined that imaging in whole epidermis, from the stratum basal to the stratum corneum, provided greater resolution per IENF. Whole epidermal imaging provides high X and Y resolution for discriminating between individual nerves. Epidermal melanocytes and keratinocytes absorb and reflect confocal laser photons and result in limited depth for imaging (~20 μm). Tissue clearing is a technique originally developed to image whole rodents, embryos, and organs with a microscope capable of volumetric imaging. To overcome light fraction in the epidermis, I optimized a version of the PACT technique to optimally clear and label isolated epidermis with IHC (Anderson and Miller, 2018b). To process volumetric imaging, I developed one script to evaluate the image stack for pixel intensity, a second script to pre-process excessively bright image stacks, and a third IENF resolving script to process and measure PGP9.5-IR objects. A method was developed to volumetrically measure and segment PGP9.5-IR objects and compare identified objects with the original (image stack) pixel intensity to visually score objects as complete IENFs, partial IENFs, IENF clumps, IENF fragments, and Langerhans cells. Size populations for small, medium, and large IENF populations were based on data for complete IENFs. Based on the frequency

distribution of IENFs for volume, surface area, and number of IENF branching points, volume and surface area appear have less variability, as a measure of size populations, than branching number, due to IR-based artifacts. The mean difference of CGRP-positive and CGRP-negative IENF fibers between groups for complete IENFs and partial IENFs was 1.8 histogram units (8-bit image, 0-255), indicating complete and partial IENF data have similar resolution. Observing and reporting qualitative (visual) data, in form of interactive 3D models, is a developing frontier. The Unreal Engine was employed to develop a digital museum for visualizing and reporting qualitative IENF morphometry data. *RTX* (requires an Nvidia RTX graphics card) and *standard* (requires a basic graphics card) versions of the IENF Museum of Living Arts are available for Windows, Apple, and Linux (https://github.com/MichaelBAnderson).

DRG neurons are highly studied for research and clinical diagnosis of peripheral neuropathy. Techniques for measuring somatic changes in DRG neurons and IENF measurements are currently useful and practiced; however, outdated. By understanding the impact of inflammation on DRG neurons at the peak swelling time point in the AIA model, we now understand DRG neurons are robustly responding to peripheral inflammation and measurements reflect neuronal response. A semi-automated method of measuring DRG neurons with IHC standardizes the protocol for identifying, measuring, and reporting error in somatosensory research. The IC-DRG algorithm is accurate, efficient, and is a powerful tool with labs and collaborative labs. I have developed the first tissue preparation protocol and algorithms to isolate, measure, and visualize individual IENFs. These advances and technologies are powerful tools and may be freely used and modified, as tools (themselves) or a foundation for entirely new projects.

CHAPTER VIII

Future Directions

The function of CGRP and the CGRP receptor components, CLR/RAMP1/RCP, are most commonly found on blood vessels and capillaries, leading to vasodilation in neuro-vascular tissue, however, the function of these receptors is less known in the neuro-avascular epidermis. To evaluate the functional capacity of CGRP in the avascular epidermis I will use a CGRP receptor antagonist to block CGRP-mediated activation in the hind paw of the rat. In the current proposal, a new class of anti-CGRP drugs, known as gepants, will be selected to evaluate the contribution of CGRP in the neuro-avascular epidermis during inflammatory sensitization. For example, the proprietary anti-CGRP antagonist Rimegepant Zydis was FDA approved (75 mg) as a dual therapy for migraines in adults (BioHavenPharmaceuticals, 2019; MedChemE, 2019). Rimegepant (BMS927711) is a small molecule gepant-class anti-CGRP drug that acts as a potent, selective, competitive and orally active antagonist on CLR and RAMP1, of the CGRP receptor complex (Pascual, 2014; Sheykhzade et al., 2017; Medkoo, 2019). IC-DRG-based image cytometry will be used for high-throughput analysis of CGRP and CGRP receptor components, CLR and RAMP1, during several time points in the acute and chronic phased of peripheral inflammation. Skin will be collected and the epidermis isolated, cleared, labeled with a pan-neuronal IENF marker, to generate healthy and disease-state morphological IENF profiles.

I hypothesize that antagonism of CGRP receptors in the epidermis will result in less inflammation and tissue damage during the acute time point of AIA-induced inflammation, potentially reducing the severity of chronic deleterious effects (chronic pain).

REFERENCES

Alekseeva OS, Gusel'nikova VV, Beznin GV, Korzhevskii DE (2015) Prospects of the nuclear protein NeuN application as an index of functional state of the vertebrate nerve cells Zh Evol Biokhim Fiziol 51:313-323.

Anderson MB (2018) Image Cytometry in Rat Dorsal Root Ganglia version 1.41 (IC-DRGs v1.41) Package: https://github.com/MichaelBAnderson/IC-DRGs_v1.41_FIJI_script. FIJI / Image J Script.

Anderson MB (2019) (Image) Pixel Intensity Variation (PIV) Package: https://github.com/MichaelBAnderson/Image_Intensity_Variation_script. ImageJ 1x script.

Anderson MB, Miller KE (2018a) Intra-Epidermal Nerve Fiber Reconstruction and Quantification in Three Dimensions. Cambridge Press: Microscopy and Microanalysis 24:1358-1359.

Anderson MB, Miller KE (2018b) Three-Dimensional Reconstruction and Quantification of PGP9.5-Immunoreactive, Sensory Intra-Epidermal Nerve Fibers Microsc Microanal 24.

Anderson MB, Miller KE, Schechter R (2010) Evaluation of rat epidermis and dermis following thermolysin separation: PGP 9.5 and Nav 1.8 localization.

Anderson MB, Das S, Miller KE (2020) Subcellular localization of neuronal nuclei (NeuN) antigen in size and calcitonin gene-related peptide (CGRP) populations of dorsal root ganglion (DRG) neurons during acute peripheral inflammation. Neuroscience Letters.

Anderson MB, Curtis JT, Miller KE (2021a) Open-source method of image cytometry in dorsal root ganglia tissue with immunofluorescence. Anal Biochem 627:114184.

Anderson MB, Das S, Miller KE (2021b) Subcellular localization of neuronal nuclei (NeuN) antigen in size and calcitonin gene-related peptide (CGRP) populations of dorsal root ganglion (DRG) neurons during acute peripheral inflammation. Neurosci Lett 760:135974.

Anderson MB, Gujar V, Miller KE, Pande R, Das S, Nawani P (2019) Separation of Rat Epidermis and Dermis with Thermolysin to Detect Site-Specific Inflammatory mRNA and Protein. Journal of Visualized Experimentation (JOVE).

Anderson MB, Gujar V, Miller KE, OPande RD, Nawani P, Subhas D (2021c) Separation of Rat Epidermis and Dermis with Thermolysin to Detect Site-Specific Inflammatory mRNA and Protein. Journal of Video Experimentation (JOVE).

Andrade R, Crisol L, Prado R, Boyano MD, Arluzea J, Arechaga J (2009) Plasma membrane and nuclear envelope integrity during the blebbing stage of apoptosis: a time-lapse study. Biol Cell 102:25-35.

Baroni A, Buommino E, De Gregorio V, Ruocco E, Ruocco V, Wolf R (2012) Structure and function of the epidermis related to barrier properties. Clin Dermatol 30:257-262.

Baruch-Eliyahu N, Rud V, Braiman A, Priel E (2019) Telomerase increasing compound protects hippocampal neurons from amyloid beta toxicity by enhancing the expression of neurotrophins and plasticity related genes. Sci Rep 9:18118.

BioHavenPharmaceuticals (2019) Rimegepant CGRP receptor antagonist.
Breese NM, George AC, Pauers LE, Stucky CL (2005) Peripheral inflammation selectively increases TRPV1 function in IB4-positive sensory neurons from adult mouse. Pain 115:37-49.
British.Pharmacological.Society (2009) Calcitonin, amylin, CGRP and adrenomedullin. British Journal of Pharmacology:S30-31.
Burg Bd, Eijsink V, Beynon RJ, Beaumont A (2004) 94 - Thermolysin and related Bacillus metallopeptidases. Handbook of Proteolytic Enzymes (Second Edition) Aspartic and Metallo Peptidases:374-387.
Burg Bd, Eisink V, Beynon RJ, Beaumont A (2013) 94 - Thermolysin and related Bacillus metallopeptidases. Handbook of Proteolytic Enzymes (Second Edition) Aspartic and Metallo Peptidases:374-387.
Cabral J, Moratti SC (2011) Hydrogels for biomedical applications. Future Med Chem 3:1877-1888.
Carpenter AE, Jones TR, Lamprecht MR, Clarke C, Kang IH, Friman O, Guertin DA, Chang JH, Lindquist RA, Moffat J, Golland P, Sabatini DM (2006) CellProfiler: image analysis software for identifying and quantifying cell phenotypes. Genome Biol 7:R100.
Chai Q, Jiao Y, Yu X (2017) Hydrogels for Biomedical Applications: Their Characteristics and the Mechanisms behind Them. Gels 3.
Charras GT (2008) A short history of blebbing. J Microsc 231:466-478.
Choi JE, Di Nardo A (2018) Skin neurogenic inflammation. Semin Immunopathol 40:249-259.
Chondrex (2017) Protocol for Adjuvant-Induced Arthritis (AIA) in Rats.
Chung K, Deisseroth K (2013) CLARITY for mapping the nervous system. Nat Methods 10:508-513.
Clayton K, Vallejo AF, Davies J, Sirvent S, Polak ME (2017) Langerhans Cells-Programmed by the Epidermis. Front Immunol 8:1676.
Collombet JM, Masqueliez C, Four E, Burckhart MF, Bernabe D, Baubichon D, Lallement G (2006) Early reduction of NeuN antigenicity induced by soman poisoning in mice can be used to predict delayed neuronal degeneration in the hippocampus. Neurosci Lett 398:337-342.
Crosby HA, Ihnat M, Miller KE (2015) Evaluating the Toxicity of the Analgesic Glutaminase Inhibitor 6-Diazo-5-Oxo-L-Norleucine in vitro and on Rat Dermal Skin Fibroblasts. MOJ Toxicol 1.
Czech E, Aksoy BA, Aksoy P, Hammerbacher J (2019) Cytokit: a single-cell analysis toolkit for high dimensional fluorescent microscopy imaging. BMC Bioinformatics 20:448.
da Silva Serra I, Husson Z, Bartlett JD, Smith ES (2016) Characterization of cutaneous and articular sensory neurons. Mol Pain 12.
Dahlhamer J, Lucas J, Zelaya C, Nahin R, Mackey S, DeBar L, Kerns R, Von Korff M, Porter L, Helmick C (2018) Prevalence of Chronic Pain and High-Impact Chronic Pain Among Adults - United States, 2016. MMWR Morb Mortal Wkly Rep 67:1001-1006.
Davoli MA, Fourtounis J, Tam J, Xanthoudakis S, Nicholson D, Robertson GS, Ng GY, Xu D (2002) Immunohistochemical and biochemical assessment of caspase-3 activation and DNA fragmentation following transient focal ischemia in the rat. Neuroscience 115:125-136.
Denk F, Bennett DL, McMahon SB (2017) Nerve Growth Factor and Pain Mechanisms. Annu Rev Neurosci 40:307-325.
Dickerson IM (2013) Role of CGRP-receptor component protein (RCP) in CLR/RAMP function. Curr Protein Pept Sci 14:407-415.
Dimauro I, Pearson T, Caporossi D, Jackson MJ (2012) A simple protocol for the subcellular fractionation of skeletal muscle cells and tissue. BMC Res Notes 5:513.

Djouhri L, Dawbarn D, Robertson A, Newton R, Lawson SN (2001) Time course and nerve growth factor dependence of inflammation-induced alterations in electrophysiological membrane properties in nociceptive primary afferent neurons. J Neurosci 21:8722-8733.

Djouhri L, Koutsikou S, Fang X, McMullan S, Lawson SN (2006) Spontaneous pain, both neuropathic and inflammatory, is related to frequency of spontaneous firing in intact C-fiber nociceptors. J Neurosci 26:1281-1292.

Du H, Hou P, Zhang W, Li Q (2018) Advances in CLARITY-based tissue clearing and imaging. Exp Ther Med 16:1567-1576.

Duan W, Zhang YP, Hou Z, Huang C, Zhu H, Zhang CQ, Yin Q (2016) Novel Insights into NeuN: from Neuronal Marker to Splicing Regulator. Mol Neurobiol 53:1637-1647.

Ebenezer GJ, Hauer P, Gibbons C, McArthur JC, Polydefkis M (2007) Assessment of epidermal nerve fibers: a new diagnostic and predictive tool for peripheral neuropathies. J Neuropathol Exp Neurol 66:1059-1073.

Eftekhari S, Warfvinge K, Blixt FW, Edvinsson L (2013) Differentiation of nerve fibers storing CGRP and CGRP receptors in the peripheral trigeminovascular system. J Pain 14:1289-1303.

Eidet JR, Pasovic L, Maria R, Jackson CJ, Utheim TP (2014) Objective assessment of changes in nuclear morphology and cell distribution following induction of apoptosis. Diagn Pathol 9:92.

Einbinder JM, Walzer RA, Mandl I (1966) Epidermal-dermal separation with proteolytic enzymes. J Invest Dermatol 46:492-504.

Eller-Smith OC, Nicol AL, Christianson JA (2018) Potential Mechanisms Underlying Centralized Pain and Emerging Therapeutic Interventions. Front Cell Neurosci 12:35.

Esposito MF, Malayil R, Hanes M, Deer T (2019) Unique Characteristics of the Dorsal Root Ganglion as a Target for Neuromodulation. Pain Med 20:S23-S30.

Fang X, Djouhri L, McMullan S, Berry C, Okuse K, Waxman SG, Lawson SN (2005) trkA is expressed in nociceptive neurons and influences electrophysiological properties via Nav1.8 expression in rapidly conducting nociceptors. J Neurosci 25:4868-4878.

Fassina G, Vita C, Dalzoppo D, Zamai M, Zambonin M, Fontana A (1986) Autolysis of thermolysin. Isolation and characterization of a folded three-fragment complex. Eur J Biochem 156:221-228.

Fehrenbacher JC, Vasko MR, Duarte DB (2012) Models of inflammation: Carrageenan- or complete Freund's Adjuvant (CFA)-induced edema and hypersensitivity in the rat. Curr Protoc Pharmacol Chapter 5:Unit5 4.

Felsher. Z (1947) Studies on the adherence of the epidermis to the corium. J Invest Dermatol

Flint MS, Dearman RJ, Kimber I, Hotchkiss SA (1998) Production and in situ localization of cutaneous tumour necrosis factor alpha (TNF-alpha) and interleukin 6 (IL-6) following skin sensitization. Cytokine 10:213-219.

Gangadharan VK, R (2013) Pain hypersensitivity mechanisms at a glance. Disease Models and Mechanisms 6:889-895.

Georg M, Fernandez-Cabada T, Bourguignon N, Karp P, Penaherrera AB, Helguera G, Lerner B, Perez MS, Mertelsmann R (2018) Development of image analysis software for quantification of viable cells in microchips. PLoS One 13:e0193605.

Glade CP, Seegers BA, Meulen EF, van Hooijdonk CA, van Erp PE, van de Kerkhof PC (1996) Multiparameter flow cytometric characterization of epidermal cell suspensions prepared from normal and hyperproliferative human skin using an optimized thermolysin-trypsin protocol. Arch Dermatol Res 288:203-210.

Gradinaru V, Treweek J, Overton K, Deisseroth K (2018) Hydrogel-Tissue Chemistry: Principles and Applications. Annu Rev Biophys 47:355-376.
Gragnani A, Sobral CS, Ferreira LM (2007) Thermolysin in human cultured keratinocyte isolation. Braz J Biol 67:105-109.
Gujar V, Miller KE (2017) Expression of nerve growth factor in adjuvant-induced arthritis (AIA): A temporal study. .
Gusel'nikova VV, Korzhevskiy DE (2015) NeuN As a Neuronal Nuclear Antigen and Neuron Differentiation Marker. Acta Naturae 7:42-47.
Haines DEA, M. D. (2002) Fundamental Neuroscience. New York: Churchhill Livingstone.
Harper AAL, S. N. (1985) Conduction velocity is related to morphological cell type in rat dorsal root ganglion neurones. Journal of Physiology:31-46.
Hay DL, Garelja ML, Poyner DR, Walker CS (2018) Update on the pharmacology of calcitonin/CGRP family of peptides: IUPHAR Review 25. Br J Pharmacol 175:3-17.
He W, Long T, Pan Q, Zhang S, Zhang Y, Zhang D, Qin G, Chen L, Zhou J (2019) Microglial NLRP3 inflammasome activation mediates IL-1beta release and contributes to central sensitization in a recurrent nitroglycerin-induced migraine model. J Neuroinflammation 16:78.
Hernandez ML, Chatlos T, Gorse KM, Lafrenaye AD (2019) Neuronal Membrane Disruption Occurs Late Following Diffuse Brain Trauma in Rats and Involves a Subpopulation of NeuN Negative Cortical Neurons. Front Neurol 10:1238.
Hoffman EM, Miller KE (2010) Peripheral inhibition of glutaminase reduces carrageenan-induced Fos expression in the superficial dorsal horn of the rat. Neurosci Lett 472:157-160.
Hoffman EM, Schechter R, Miller KE (2010) Fixative composition alters distributions of immunoreactivity for glutaminase and two markers of nociceptive neurons, Nav1.8 and TRPV1, in the rat dorsal root ganglion. J Histochem Cytochem 58:329-344.
Hoffman EM, Zhang Z, Schechter R, Miller KE (2016) Glutaminase Increases in Rat Dorsal Root Ganglion Neurons after Unilateral Adjuvant-Induced Hind Paw Inflammation. Biomolecules 6:10.
Hoffman EM, Zhang Z, Anderson MB, Schechter R, Miller KE (2011) Potential mechanisms for hypoalgesia induced by anti-nerve growth factor immunoglobulin are identified using autoimmune nerve growth factor deprivation. Neuroscience 193:452-465.
Hsieh ST, Choi S, Lin WM, Chang YC, McArthur JC, Griffin JW (1996) Epidermal denervation and its effects on keratinocytes and Langerhans cells. J Neurocytol 25:513-524.
Hu X, Jiang Z, Liu N (2006) A novel approach for harvesting lymphatic endothelial cells from human foreskin dermis. Lymphat Res Biol 4:191-198.
Hybbinette S, Bostrom M, Lindberg K (1999) Enzymatic dissociation of keratinocytes from human skin biopsies for in vitro cell propagation. Exp Dermatol 8:30-38.
Ibitokun BO, Anderson MB, Miller KE (2010) Separation of corneal epithelium from the stroma using thermolysin: evaluation of corneal afferents.
Iyengar S, Ossipov MH, Johnson KW (2017) The role of calcitonin gene-related peptide in peripheral and central pain mechanisms including migraine. Pain 158:543-559.
Ji RR, Samad TA, Jin SX, Schmoll R, Woolf CJ (2002) p38 MAPK activation by NGF in primary sensory neurons after inflammation increases TRPV1 levels and maintains heat hyperalgesia. Neuron 36:57-68.
Jian L, Cao Y, Zou Y (2020) Dermal-Epidermal Separation by Enzyme. Methods Mol Biol 2109:27-30.
Johannes CB, Le TK, Zhou X, Johnston JA, Dworkin RH (2010) The prevalence of chronic pain in United States adults: results of an Internet-based survey. J Pain 11:1230-1239.

Kamentsky L, Jones TR, Fraser A, Bray MA, Logan DJ, Madden KL, Ljosa V, Rueden C, Eliceiri KW, Carpenter AE (2011) Improved structure, function and compatibility for CellProfiler: modular high-throughput image analysis software. Bioinformatics 27:1179-1180.
Karsan N, Goadsby PJ (2015) CGRP mechanism antagonists and migraine management. Curr Neurol Neurosci Rep 15:25.
Kawamura Y, Dyck PJ (1978) Evidence for three populations by size in L5 spinal ganglion in man. J Neuropathol Exp Neurol 37:269-272.
Kennedy J, Roll JM, Schraudner T, Murphy S, McPherson S (2014) Prevalence of persistent pain in the U.S. adult population: new data from the 2010 national health interview survey. J Pain 15:979-984.
Kim KK, Adelstein RS, Kawamoto S (2009) Identification of neuronal nuclei (NeuN) as Fox-3, a new member of the Fox-1 gene family of splicing factors. J Biol Chem 284:31052-31061.
Kim KK, Kim YC, Adelstein RS, Kawamoto S (2011) Fox-3 and PSF interact to activate neural cell-specific alternative splicing. Nucleic Acids Res 39:3064-3078.
Klionsky DJ et al. (2016) Guidelines for the use and interpretation of assays for monitoring autophagy (3rd edition). Autophagy 12:1-222.
Kresge N, Simoni RD, Hill RL (2009) Structural Studies of Thermolysin: the Work of Brian W. Matthews. Journal of Biological Chemistry:e8-e9.
Lauria GB, D. R.; Johansson, O; McArthur, JC; Mellgren, S. I.; Nolano, M; Rosenberg, N; Sommer, C.; European Federation of Neurological Societies. (2005) EFNS guidelines on the use of skin biopsy in the diagnosis of peripheral neuropathy. European Journal of Neurology:747-758.
Lawson SN, Biscoe TJ (1979) Development of mouse dorsal root ganglia: an autoradiographic and quantitative study. J Neurocytol 8:265-274.
Lawson SN, Harper AA, Harper EI, Garson JA, Anderton BH (1984) A monoclonal antibody against neurofilament protein specifically labels a subpopulation of rat sensory neurones. J Comp Neurol 228:263-272.
Li KK, Shen SS, Deng X, Shiu HT, Siu WS, Leung PC, Ko CH, Cheng BH (2018) Dihydrofisetin exerts its anti-inflammatory effects associated with suppressing ERK/p38 MAPK and Heme Oxygenase-1 activation in lipopolysaccharide-stimulated RAW 264.7 macrophages and carrageenan-induced mice paw edema. Int Immunopharmacol 54:366-374.
Lin YT, Ro LS, Wang HL, Chen JC (2011) Up-regulation of dorsal root ganglia BDNF and trkB receptor in inflammatory pain: an in vivo and in vitro study. J Neuroinflammation 8:126.
Ma C, Shu Y, Zheng Z, Chen Y, Yao H, Greenquist KW, White FA, LaMotte RH (2003) Similar electrophysiological changes in axotomized and neighboring intact dorsal root ganglion neurons. J Neurophysiol 89:1588-1602.
Ma X, Turnbull PC, Crapper EP, Wang H, Drannik A, Jiang F, Xia S, Turnbull J (2016) Cytosolic localization of Fox proteins in motor neurons of G93A SOD1 mice. Histochem Cell Biol 145:545-559.
Mandelkow R, Gumbel D, Ahrend H, Kaul A, Zimmermann U, Burchardt M, Stope MB (2017) Detection and Quantification of Nuclear Morphology Changes in Apoptotic Cells by Fluorescence Microscopy and Subsequent Analysis of Visualized Fluorescent Signals. Anticancer Res 37:2239-2244.
Mangus LM, Rao DB, Ebenezer GJ (2020) Intraepidermal Nerve Fiber Analysis in Human Patients and Animal Models of Peripheral Neuropathy: A Comparative Review. Toxicol Pathol 48:59-70.
Manti S, Brown P, Perez MK, Piedimonte G (2017) The Role of Neurotrophins in Inflammation and Allergy. Vitam Horm 104:313-341.

Martin SL, Reid AJ, Verkhratsky A, Magnaghi V, Faroni A (2019) Gene expression changes in dorsal root ganglia following peripheral nerve injury: roles in inflammation, cell death and nociception. Neural Regen Res 14:939-947.
McPhail LT, McBride CB, McGraw J, Steeves JD, Tetzlaff W (2004) Axotomy abolishes NeuN expression in facial but not rubrospinal neurons. Exp Neurol 185:182-190.
MedChemE (2019) Rimegepant (CGRP antagonist). In. https://www.medchemexpress.com/BMS-927711.html.
Medkoo (2019) Rimegepant.
Michel M, L'Heureux N, Pouliot R, Xu W, Auger FA, Germain L (1999) Characterization of a new tissue-engineered human skin equivalent with hair. In Vitro Cell Dev Biol Anim 35:318-326.
Miller KE, Douglas VD, Kaneko T (1993) Glutaminase immunoreactive neurons in the rat dorsal root ganglion contain calcitonin gene-related peptide (CGRP). Neurosci Lett 160:113-116.
Miller KE, Balbas JC, Benton RL, Lam TS, Edwards KM, Kriebel RM, Schechter R (2012) Glutaminase immunoreactivity and enzyme activity is increased in the rat dorsal root ganglion following peripheral inflammation. Pain Res Treat 2012:414697.
Mullen RJ, Buck CR, Smith AM (1992) NeuN, a neuronal specific nuclear protein in vertebrates. Development 116:201-211.
Nawani P, Anderson MB, Miller KE (2011) Structure-property relationship of skin.
Pascual J (2014) Efficacy of BMS-927711 and other gepants vs triptans: there seem to be other players besides CGRP. Cephalalgia 34:1028-1029.
Pereira MP, Muhl S, Pogatzki-Zahn EM, Agelopoulos K, Stander S (2016) Intraepidermal Nerve Fiber Density: Diagnostic and Therapeutic Relevance in the Management of Chronic Pruritus: a Review. Dermatol Ther (Heidelb) 6:509-517.
Petho G, Reeh PW (2012) Sensory and signaling mechanisms of bradykinin, eicosanoids, platelet-activating factor, and nitric oxide in peripheral nociceptors. Physiol Rev 92:1699-1775.
Petrilli G, Fisogni S, Rosai J, Festa S, Torri L, Rodeschi A, Licini S, Facchetti F (2017) Nuclear Bubbles (Nuclear Pseudo-Pseudoinclusions): A Pitfall in the Interpretation of Microscopic Sections From the Thyroid and Other Human Organs. Am J Surg Pathol 41:140-141.
Portiansky EL, Barbeito CG, Gimeno EJ, Zuccolilli GO, Goya RG (2006) Loss of NeuN immunoreactivity in rat spinal cord neurons during aging. Exp Neurol 202:519-521.
Prado MA, Evans-Bain B, Dickerson IM (2002) Receptor component protein (RCP): a member of a multi-protein complex required for G-protein-coupled signal transduction. Biochem Soc Trans 30:460-464.
Qu L, Caterina MJ (2016) Enhanced excitability and suppression of A-type K(+) currents in joint sensory neurons in a murine model of antigen-induced arthritis. Sci Rep 6:28899.
R Core T (2013) R: A language and environment for statistical

computing. R Foundation for Statistical Computing.
Rakhorst HA, Tra WM, Posthumus-van Sluijs SJ, de Groot E, van Osch GJ, van Neck JW, Hofer SO (2006) Mucosal keratinocyte isolation: a short comparative study on thermolysin and dispase. Int J Oral Maxillofac Surg 35:935-940.
Reynolds AR, Saunders MA, Prendergast MA (2016) Ethanol Stimulates Endoplasmic Reticulum Inositol Triphosphate and Sigma Receptors to Promote Withdrawal-Associated Loss of Neuron-Specific Nuclear Protein/Fox-3. Alcohol Clin Exp Res 40:1454-1461.

Ririe DG, Liu B, Clayton B, Tong C, Eisenach JC (2008) Electrophysiologic characteristics of large neurons in dorsal root ganglia during development and after hind paw incision in the rat. Anesthesiology 109:111-117.
Russell FA, King R, Smillie SJ, Kodji X, Brain SD (2014) Calcitonin gene-related peptide: physiology and pathophysiology. Physiol Rev 94:1099-1142.
Sajja VS, Ereifej ES, VandeVord PJ (2014) Hippocampal vulnerability and subacute response following varied blast magnitudes. Neurosci Lett 570:33-37.
Santoianni RA, Hammami A (2013) Nuclear Bubbling: An Overlooked Artifact. Journal of Histotechnology 13:135-136.
Sato JD, Kan M (2001) Media for culture of mammalian cells. Curr Protoc Cell Biol Chapter 1:Unit 1 2.
Schakel K, Schon MP, Ghoreschi K (2016) [Pathogenesis of psoriasis]. Hautarzt 67:422-431.
Schindelin J, Arganda-Carreras I, Frise E, Kaynig V, Longair M, Pietzsch T, Preibisch S, Rueden C, Saalfeld S, Schmid B, Tinevez JY, White DJ, Hartenstein V, Eliceiri K, Tomancak P, Cardona A (2012) Fiji: an open-source platform for biological-image analysis. Nat Methods 9:676-682.
Schneider CA, Rasband WS, Eliceiri KW (2012) NIH Image to ImageJ: 25 years of image analysis. Nat Methods 9:671-675.
Schou WS, Ashina S, Amin FM, Goadsby PJ, Ashina M (2017) Calcitonin gene-related peptide and pain: a systematic review. J Headache Pain 18:34.
Sheykhzade M, Amandi N, Pla MV, Abdolalizadeh B, Sams A, Warfvinge K, Edvinsson L, Pickering DS (2017) Binding and functional pharmacological characteristics of gepant-type antagonists in rat brain and mesenteric arteries. Vascul Pharmacol 90:36-43.
Singer C (1914) Notes on the Early History of Microscopy. Proc R Soc Med 7:247-279.
Stroke NIoNDa (2021) NINDS.NIH.gov.
Suzuki H, Aoyama Y, Senzaki K, Vincler M, Wittenauer S, Yoshikawa M, Ozaki S, Oppenheim RW, Shiga T (2010) Characterization of sensory neurons in the dorsal root ganglia of Bax-deficient mice. Brain Res 1362:23-31.
Timar B, Popescu S, Timar R, Baderca F, Duica B, Vlad M, Levai C, Balinisteanu B, Simu M (2016) The usefulness of quantifying intraepidermal nerve fibers density in the diagnostic of diabetic peripheral neuropathy: a cross-sectional study. Diabetol Metab Syndr 8:31.
Tschachler E, Reinisch CM, Mayer C, Paiha K, Lassmann H, Weninger W (2004) Sheet preparations expose the dermal nerve plexus of human skin and render the dermal nerve end organ accessible to extensive analysis. J Invest Dermatol 122:177-182.
Unal-Cevik I, Kilinc M, Gursoy-Ozdemir Y, Gurer G, Dalkara T (2004) Loss of NeuN immunoreactivity after cerebral ischemia does not indicate neuronal cell loss: a cautionary note. Brain Res 1015:169-174.
Van Nassauw L, Wu M, De Jonge F, Adriaensen D, Timmermans JP (2005) Cytoplasmic, but not nuclear, expression of the neuronal nuclei (NeuN) antibody is an exclusive feature of Dogiel type II neurons in the guinea-pig gastrointestinal tract. Histochem Cell Biol 124:369-377.
Walzer C, Benathan M, Frenk E (1989) Thermolysin treatment: a new method for dermo-epidermal separation. J Invest Dermatol 92:78-81.
Wang T, Miller KE (2016) Characterization of glutamatergic neurons in the rat atrial intrinsic cardiac ganglia that project to the cardiac ventricular wall. Neuroscience 329:134-150.
Watershed A (2019) ImageJ Documentation Wiki: https://imagejdocu.tudor.lu/doku.php?id=plugin:segmentation:adjustable_watershed:start. In.

Weston C, Winfield I, Harris M, Hodgson R, Shah A, Dowell SJ, Mobarec JC, Woodlock DA, Reynolds CA, Poyner DR, Watkins HA, Ladds G (2016) Receptor Activity-modifying Protein-directed G Protein Signaling Specificity for the Calcitonin Gene-related Peptide Family of Receptors. J Biol Chem 291:21925-21944.

Whiteley PE, Dalrymple SA (2001) Models of inflammation: adjuvant-induced arthritis in the rat. Curr Protoc Pharmacol Chapter 5:Unit5 5.

Woo J, Lee M, Seo JM, Park HS, Cho YE (2016) Optimization of the optical transparency of rodent tissues by modified PACT-based passive clearing. Exp Mol Med 48:e274.

Woolf CJ, Ma Q (2007) Nociceptors--noxious stimulus detectors. Neuron 55:353-364.

Wu KL, Li YQ, Tabassum A, Lu WY, Aubert I, Wong CS (2010) Loss of neuronal protein expression in mouse hippocampus after irradiation. J Neuropathol Exp Neurol 69:272-280.

Yang B, Treweek JB, Kulkarni RP, Deverman BE, Chen CK, Lubeck E, Shah S, Cai L, Gradinaru V (2014) Single-cell phenotyping within transparent intact tissue through whole-body clearing. Cell 158:945-958.

Yousef H, Alhajj M, Sharma S (2021) Anatomy, Skin (Integument), Epidermis.

Yufera A, Gallego E, Molina J (2011) ImagCell: a computer tool for cell culture image processing applications in bioimpedance measurements. Adv Exp Med Biol 696:733-740.

Zhang Z (2013) The Role of Dorsal Root Ganglion Glutaminase in Acute and Chronic Inflammatory Pain. PhD Dissertation, Oklahoma State University, Stillwater, OK, USA, 2013.

Zou Y, Maibach HI (2018) Dermal-epidermal separation methods: research implications. Arch Dermatol Res 310:1-9.

Appendix A – IC-DRG: EMGI Scoring Script

This script is operational under ImageJ/FIJI (version v1.53f +)

script begins here:

"// - - - This script evaluates NeuN (neuCyto) and DAPI (neuNuc) markers to determine optimal camera exposure when taking images for processing in IC-DRG (v1.41)

//----------

// - - - Average resulting MGI from EMGIS for all NeuN images and that number is the NeuN EMGI Score

//----------

// - - - Average resulting MGI from EMGIS for all DAPI images and that number is the DAPI EMGI Score//----------

// - - - - - - Use EMGI Scoring Chart to determine if camera exposure optimization is needed

```
//-----------------------------------------------------------------------------------------------------------

// - - - - - - - - - - - Once in the Optimal Zone of the EMGIS chart for both NeuN and
DAPI, run through IC-DRG (1.41)

//   version 1:  Set-up foundation

//   version 2:  Trying to get saving to work properly.  I would like one file with all
measurements but I am getting separate files.  I have tried to compartmentilize the
saveing feature in a third macro in this version.

//   version 3:  This version has fixed all of the issues.  If the user experiences dpulicate
data in "Results.csv" then it is because there was already data in the Results window.  If
you experience this then select all data in Results window and clear, then run.  I tried to
do this with a macro but there is no function for doing this in Results window.

//   version 0.04:  Duplicate of version 3.  Working Version

#@ File (label = "Select folder input - DAPI", style = "directory")
DAPI_Evaluation_input

#@ File (label = "Select folder input - NeuN", style = "directory")
NeuN_Evaluation_input

#@ File (label = "Select folder input - Measurement", style = "directory")
Measurement_output

#@ String (label = "File suffix", value = ".tif") suffix
```

```
//-------------------------------------------------------------------------------------------------
NeuN Evaluation
setBatchMode(true);
processFolder1(NeuN_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder1 (NeuN_Evaluation_input)
}
function processFile1 (NeuN_Evaluation_input, file) {
  open(NeuN_Evaluation_input + File.separator + file);
setAutoThreshold("Default dark");
//run("Threshold...");
setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
}
close("*");
```

```
//-------------------------------------------------------------------------------------------------

DAPI Evaluation

setBatchMode(true);

processFolder2(DAPI_Evaluation_input);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder2 (DAPI_Evaluation_input) {

  list = getFileList(DAPI_Evaluation_input);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))

              processFolder2(DAPI_Evaluation_input + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile2(DAPI_Evaluation_input, Measurement_output, list[i]);

  }

}

function processFile2 (DAPI_Evaluation_input, Measurement_output, file) {

  open(DAPI_Evaluation_input + File.separator + file);

setAutoThreshold("Default dark");
```

```
//run("Threshold...");

setThreshold(0, 30);
                        // <----    Threshold value to eliminate background in MGI
Evaluation

run("Set Measurements...", "mean display redirect=None decimal=3");

run("Measure");

}

  saveAs("Measurements", Measurement_output + File.separator  + "Results.csv");

selectWindow("Results");

run("Close");

//------------------------------------------------------------------------------------------------------
CLOSE

beep();

Dialog.create("Finished");

Dialog.addMessage("         All Images Have Been Successfully Converted\n"

+"

                                \n"

+"

                                   \n"
```

```
+"                                  ____                \n"
+"                                  |    |              \n"
+"                                  ====                \n"
+"                                  |    |__            \n"
+"                                  |    |-. \\ \\       \n"
+"                                  |__|    \\ \\        \n"
+"                                   | |       ||      \n"
+"                                  ====  __|           \n"
+"                                _________ ||__        \n"
+"                              /__ Miller  Lab __\\      \n"
+"                                                      \n"
+"
                                    \n"
+"
                                  \n"
+"                 MGI Evaluation for NueN and DAPI is complete                 \n"
+"
                                  ");
```

Dialog.show();”

Appendix B – IC-DRG: Pixel Intensity Evaluation Script

This script is operational under ImageJ/FIJI (version v1.53f +)

script begins here:

"// version 1: Set-up foundation

// version 2: Trying to get saving to work properly. I would like one file with all measurements but I am getting separate files. I have tried to compartmentilize the saveing feature in a third macro in this version.

// version 3: This version has fixed all of the issues. If the user experiences dupulicate data in "Results.csv" then it is because there was already data in the Results window. If you experience this then select all data in Results window and clear, then run. I tried to do this with a macro but there is no function for doing this in Results window.

// version 0.04: Duplicate of version 3. Working Version

#@ File (label = "folder input - DAPI", style = "directory") DAPI_Evaluation_input

#@ File (label = "folder input - NeuN", style = "directory") NeuN_Evaluation_input

#@ File (label = "NeuN output: -120", style = "directory")
NeuN_Measurement_output_minus120

#@ File (label = "NeuN output: -100", style = "directory")
NeuN_Measurement_output_minus100

#@ File (label = "NeuN output: -80", style = "directory")

NeuN_Measurement_output_minus80

#@ File (label = "NeuN output: -60", style = "directory")

NeuN_Measurement_output_minus60

#@ File (label = "NeuN output: -40", style = "directory")

NeuN_Measurement_output_minus40

#@ File (label = "NeuN output: -20", style = "directory")

NeuN_Measurement_output_minus20

#@ File (label = "NeuN output: +20", style = "directory")

NeuN_Measurement_output_plus20

#@ File (label = "NeuN output: +40", style = "directory")

NeuN_Measurement_output_plus40

#@ File (label = "NeuN output: +60", style = "directory")

NeuN_Measurement_output_plus60

#@ File (label = "NeuN output: +80", style = "directory")

NeuN_Measurement_output_plus80

#@ File (label = "NeuN output: +100", style = "directory")

NeuN_Measurement_output_plus100

#@ File (label = "NeuN output: +120", style = "directory")

NeuN_Measurement_output_plus120

```
#@ File (label = "DAPI output: -120", style = "directory")
DAPI_Measurement_output_minus120

#@ File (label = "DAPI output: -100", style = "directory")
DAPI_Measurement_output_minus100

#@ File (label = "DAPI output: -80", style = "directory")
DAPI_Measurement_output_minus80

#@ File (label = "DAPI output: -60", style = "directory")
DAPI_Measurement_output_minus60

#@ File (label = "DAPI output: -40", style = "directory")
DAPI_Measurement_output_minus40

#@ File (label = "DAPI output: -20", style = "directory")
DAPI_Measurement_output_minus20

#@ File (label = "DAPI output: +20", style = "directory")
DAPI_Measurement_output_plus20

#@ File (label = "DAPI output: +40", style = "directory")
DAPI_Measurement_output_plus40

#@ File (label = "DAPI output: +60", style = "directory")
DAPI_Measurement_output_plus60

#@ File (label = "DAPI output: +80", style = "directory")
DAPI_Measurement_output_plus80
```

```
#@ File (label = "DAPI output: +100", style = "directory")
DAPI_Measurement_output_plus100

#@ File (label = "DAPI output: +120", style = "directory")
DAPI_Measurement_output_plus120

#@ String (label = "File suffix", value = ".tif") suffix

//-----------------------------------------------------------------------------------------------------
NeuN Evaluation: -120 MGI

setBatchMode(true);

processFolder1(NeuN_Evaluation_input);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder1 (NeuN_Evaluation_input) {

  list = getFileList(NeuN_Evaluation_input);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))

                processFolder1(NeuN_Evaluation_input + File.separator + list[i]);

        if(endsWith(list[i], suffix))

                processFile1(NeuN_Evaluation_input, list[i]);
```

```
  }
}
function processFile1 (NeuN_Evaluation_input, file) {
  open(NeuN_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(120, 375);              // adjusted brightness and contrast by -120
run("Apply LUT");
setAutoThreshold("Default dark");
//run("Threshold...");
setThreshold(0, 30);
                                                    //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", NeuN_Measurement_output_minus120 + File.separator + file);
  saveAs("Measurements", NeuN_Measurement_output_minus120 + File.separator  +
"Results.csv");
}
```

```
selectWindow("Results");

run("Close");

//-------------------------------------------------------------------------------------------------

NeuN Evaluation: -100 MGI

setBatchMode(true);

processFolder2(NeuN_Evaluation_input);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder2 (NeuN_Evaluation_input) {

  list = getFileList(NeuN_Evaluation_input);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))

                processFolder2(NeuN_Evaluation_input + File.separator + list[i]);

        if(endsWith(list[i], suffix))

                processFile2(NeuN_Evaluation_input, list[i]);

  }

}

function processFile2 (NeuN_Evaluation_input, file) {
```

```
  open(NeuN_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(100, 355);              // adjusted brightness and contrast by -100
run("Apply LUT");
setAutoThreshold("Default dark");
//run("Threshold...");
setThreshold(0, 30);
                                                    //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", NeuN_Measurement_output_minus100 + File.separator + file);
  saveAs("Measurements", NeuN_Measurement_output_minus100 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
```

```
//-------------------------------------------------------------------------------------------------------
NeuN Evaluation: -80 MGI
setBatchMode(true);
processFolder3(NeuN_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder3 (NeuN_Evaluation_input) {
  list = getFileList(NeuN_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))
              processFolder3(NeuN_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile3(NeuN_Evaluation_input, list[i]);
  }
}
function processFile3 (NeuN_Evaluation_input, file) {
  open(NeuN_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
```

```
setMinAndMax(80, 335);                  // adjusted brightness and contrast by -80

run("Apply LUT");

setAutoThreshold("Default dark");

//run("Threshold...");

setThreshold(0, 30);
                                                        //  <----    Threshold value to
eliminate background in MGI Evaluation

run("Set Measurements...", "mean display redirect=None decimal=3");

run("Measure");

  saveAs("Tiff", NeuN_Measurement_output_minus80 + File.separator + file);

  saveAs("Measurements", NeuN_Measurement_output_minus80 + File.separator  +
"Results.csv");

}

selectWindow("Results");

run("Close");

//-------------------------------------------------------------------------------------------------------
NeuN Evaluation: -60 MGI

setBatchMode(true);
```

```
processFolder4(NeuN_Evaluation_input);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder4 (NeuN_Evaluation_input) {

  list = getFileList(NeuN_Evaluation_input);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))

              processFolder4(NeuN_Evaluation_input + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile4(NeuN_Evaluation_input, list[i]);

  }

}

function processFile4 (NeuN_Evaluation_input, file) {

  open(NeuN_Evaluation_input + File.separator + file);




//run("Brightness/Contrast...");
```

```
setMinAndMax(60, 315);                  // adjusted brightness and contrast by -60

run("Apply LUT");

setAutoThreshold("Default dark");

//run("Threshold...");

setThreshold(0, 30);
                                                        //  <----    Threshold value to
eliminate background in MGI Evaluation

run("Set Measurements...", "mean display redirect=None decimal=3");

run("Measure");

  saveAs("Tiff", NeuN_Measurement_output_minus60 + File.separator + file);

  saveAs("Measurements", NeuN_Measurement_output_minus60 + File.separator  +
"Results.csv");

}

selectWindow("Results");

run("Close");

//-------------------------------------------------------------------------------------------------------
NeuN Evaluation: -40 MGI

setBatchMode(true);
```

```
processFolder5(NeuN_Evaluation_input);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder5 (NeuN_Evaluation_input) {

  list = getFileList(NeuN_Evaluation_input);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))

              processFolder5(NeuN_Evaluation_input + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile5(NeuN_Evaluation_input, list[i]);

  }

}

function processFile5 (NeuN_Evaluation_input, file) {

  open(NeuN_Evaluation_input + File.separator + file);

//run("Brightness/Contrast...");

setMinAndMax(40, 275);              // adjusted brightness and contrast by -40

run("Apply LUT");
```

```
//run("Threshold...");
setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", NeuN_Measurement_output_minus40 + File.separator + file);
  saveAs("Measurements", NeuN_Measurement_output_minus40 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------
NeuN Evaluation: -20 MGI
setBatchMode(true);
processFolder6(NeuN_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder6 (NeuN_Evaluation_input) {
```

```
  list = getFileList(NeuN_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))
              processFolder6(NeuN_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile6(NeuN_Evaluation_input, list[i]);
  }
}
function processFile6 (NeuN_Evaluation_input, file) {
  open(NeuN_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(20, 275);              // adjusted brightness and contrast by -20
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                  // <----  Threshold value to
eliminate background in MGI Evaluation
```

```
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", NeuN_Measurement_output_minus20 + File.separator + file);
  saveAs("Measurements", NeuN_Measurement_output_minus20 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------
NeuN Evaluation: +20 MGI
setBatchMode(true);
processFolder8(NeuN_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder8 (NeuN_Evaluation_input) {
  list = getFileList(NeuN_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))
```

```
                processFolder8(NeuN_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
                processFile8(NeuN_Evaluation_input, list[i]);
  }
}
function processFile8 (NeuN_Evaluation_input, file) {
  open(NeuN_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 213);                    // adjusted brightness and contrast by +20
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                    //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", NeuN_Measurement_output_plus20 + File.separator + file);
```

```
  saveAs("Measurements", NeuN_Measurement_output_plus20 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------------
NeuN Evaluation: +40 MGI
setBatchMode(true);
processFolder10(NeuN_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder10 (NeuN_Evaluation_input) {
  list = getFileList(NeuN_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))
              processFolder10(NeuN_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile10(NeuN_Evaluation_input, list[i]);
```

```
  }
}
function processFile10 (NeuN_Evaluation_input, file) {
  open(NeuN_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 173);                    // adjusted brightness and contrast by +40
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                    //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", NeuN_Measurement_output_plus40 + File.separator + file);
  saveAs("Measurements", NeuN_Measurement_output_plus40 + File.separator  +
"Results.csv");
}
selectWindow("Results");
```

```
run("Close");

//-----------------------------------------------------------------------------------------------------

NeuN Evaluation: +60 MGI

setBatchMode(true);

processFolder12(NeuN_Evaluation_input);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder12 (NeuN_Evaluation_input) {

  list = getFileList(NeuN_Evaluation_input);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))

              processFolder12(NeuN_Evaluation_input + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile12(NeuN_Evaluation_input, list[i]);

  }

}

function processFile12 (NeuN_Evaluation_input, file) {

  open(NeuN_Evaluation_input + File.separator + file);
```

```
//run("Brightness/Contrast...");
setMinAndMax(0, 133);                    // adjusted brightness and contrast by +60
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                    //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", NeuN_Measurement_output_plus60 + File.separator + file);
  saveAs("Measurements", NeuN_Measurement_output_plus60 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------------
NeuN Evaluation: +80 MGI
setBatchMode(true);
```

```
processFolder13(NeuN_Evaluation_input);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder13 (NeuN_Evaluation_input) {

  list = getFileList(NeuN_Evaluation_input);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))

              processFolder13(NeuN_Evaluation_input + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile13(NeuN_Evaluation_input, list[i]);

  }

}

function processFile13 (NeuN_Evaluation_input, file) {

  open(NeuN_Evaluation_input + File.separator + file);

//run("Brightness/Contrast...");

setMinAndMax(0, 93);                  // adjusted brightness and contrast by +60

run("Apply LUT");
```

```
//run("Threshold...");
setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", NeuN_Measurement_output_plus80 + File.separator + file);
  saveAs("Measurements", NeuN_Measurement_output_plus80 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------
NeuN Evaluation: +100 MGI
setBatchMode(true);
processFolder14(NeuN_Evaluation_input);


// function to scan folders/subfolders/files to find files with correct suffix
```

```
function processFolder14 (NeuN_Evaluation_input) {
  list = getFileList(NeuN_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))
              processFolder14(NeuN_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile14(NeuN_Evaluation_input, list[i]);
  }
}
function processFile14 (NeuN_Evaluation_input, file) {
  open(NeuN_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 53);                    // adjusted brightness and contrast by +60
run("Apply LUT");
//run("Threshold...");
```

```
setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", NeuN_Measurement_output_plus100 + File.separator + file);
  saveAs("Measurements", NeuN_Measurement_output_plus100 + File.separator  +
"Results.csv");
}


selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------
NeuN Evaluation: +120 MGI
setBatchMode(true);
processFolder15(NeuN_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder15 (NeuN_Evaluation_input) {
```

```
  list = getFileList(NeuN_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(NeuN_Evaluation_input + File.separator + list[i]))
              processFolder15(NeuN_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile15(NeuN_Evaluation_input, list[i]);
  }
}
function processFile15 (NeuN_Evaluation_input, file) {
  open(NeuN_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 13);                    // adjusted brightness and contrast by +60
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                  //  <----    Threshold value to
eliminate background in MGI Evaluation
```

```
run("Set Measurements...", "mean display redirect=None decimal=3");

run("Measure");

  saveAs("Tiff", NeuN_Measurement_output_plus120 + File.separator + file);

  saveAs("Measurements", NeuN_Measurement_output_plus120 + File.separator  +
"Results.csv");

}

selectWindow("Results");

run("Close");

//-------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------COMPLETION OF NEUN
(neuCyto) PROCESSING ABOVE

//-------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------

//-------------------------------------------------------------------------------------------------------
-----------------------------------DAPI (neuNuc) PROCESSING BELOW

//-------------------------------------------------------------------------------------------------------
DAPI Evaluation: -120 MGI

setBatchMode(true);

processFolder16(DAPI_Evaluation_input);
```

```
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder16 (DAPI_Evaluation_input) {
  list = getFileList(DAPI_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
              processFolder16(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile16(DAPI_Evaluation_input, list[i]);
  }
}
function processFile16 (DAPI_Evaluation_input, file) {
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(120, 375);              // adjusted brightness and contrast by -120
run("Apply LUT");
setAutoThreshold("Default dark");
```

```
//run("Threshold...");
setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_minus120 + File.separator + file);
  saveAs("Measurements", DAPI_Measurement_output_minus120 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------
DAPI Evaluation: -100 MGI
setBatchMode(true);
processFolder17(DAPI_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder17 (DAPI_Evaluation_input) {
```

```
  list = getFileList(DAPI_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
              processFolder17(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile17(DAPI_Evaluation_input, list[i]);
  }
}
function processFile17 (DAPI_Evaluation_input, file) {
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(100, 355);          // adjusted brightness and contrast by -100
run("Apply LUT");
setAutoThreshold("Default dark");
//run("Threshold...");
```

```
setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_minus100 + File.separator + file);
  saveAs("Measurements", DAPI_Measurement_output_minus100 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------
DAPI Evaluation: -80 MGI
setBatchMode(true);
processFolder18(DAPI_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder18 (DAPI_Evaluation_input) {
  list = getFileList(DAPI_Evaluation_input);
```

```
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
              processFolder18(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile18(DAPI_Evaluation_input, list[i]);
  }
}
function processFile18 (DAPI_Evaluation_input, file) {
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(80, 335);              // adjusted brightness and contrast by -80
run("Apply LUT");
setAutoThreshold("Default dark");
//run("Threshold...");
setThreshold(0, 30);
                                          // <----   Threshold value to
eliminate background in MGI Evaluation
```

```
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_minus80 + File.separator + file);
  saveAs("Measurements", DAPI_Measurement_output_minus80 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------------
DAPI Evaluation: -60 MGI
setBatchMode(true);
processFolder19(DAPI_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder19 (DAPI_Evaluation_input) {
  list = getFileList(DAPI_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
```

```
                processFolder19(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
                processFile19(DAPI_Evaluation_input, list[i]);
  }
}
function processFile19 (DAPI_Evaluation_input, file) {
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(60, 315);              // adjusted brightness and contrast by -60
run("Apply LUT");
setAutoThreshold("Default dark");
//run("Threshold...");
setThreshold(0, 30);
                                                    // <----  Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_minus60 + File.separator + file);
```

```
  saveAs("Measurements", DAPI_Measurement_output_minus60 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------------
DAPI Evaluation: -40 MGI


setBatchMode(true);
processFolder20(DAPI_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder20 (DAPI_Evaluation_input) {
  list = getFileList(DAPI_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
              processFolder20(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
```

```
                processFile20(DAPI_Evaluation_input, list[i]);
    }
}
function processFile20 (DAPI_Evaluation_input, file) {
   open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(40, 275);                  // adjusted brightness and contrast by -40
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                    //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
   saveAs("Tiff", DAPI_Measurement_output_minus40 + File.separator + file);
   saveAs("Measurements", DAPI_Measurement_output_minus40 + File.separator  +
"Results.csv");
}
```

```
selectWindow("Results");

run("Close");

//-------------------------------------------------------------------------------------------------------

DAPI Evaluation: -20 MGI

setBatchMode(true);

processFolder21(DAPI_Evaluation_input);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder21 (DAPI_Evaluation_input) {

  list = getFileList(DAPI_Evaluation_input);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))

                processFolder21(DAPI_Evaluation_input + File.separator + list[i]);

        if(endsWith(list[i], suffix))

                processFile21(DAPI_Evaluation_input, list[i]);

  }

}

function processFile21 (DAPI_Evaluation_input, file) {
```

```
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(20, 275);                // adjusted brightness and contrast by -20
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_minus20 + File.separator + file);
  saveAs("Measurements", DAPI_Measurement_output_minus20 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------------
DAPI Evaluation: +20 MGI
```

```
setBatchMode(true);
processFolder22(DAPI_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder22 (DAPI_Evaluation_input) {
  list = getFileList(DAPI_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
              processFolder22(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile22(DAPI_Evaluation_input, list[i]);
  }
}
function processFile22 (DAPI_Evaluation_input, file) {
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 214);              // adjusted brightness and contrast by +10
```

```
run("Apply LUT");

//run("Threshold...");

setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation

run("Set Measurements...", "mean display redirect=None decimal=3");

run("Measure");

  saveAs("Tiff", DAPI_Measurement_output_plus20 + File.separator + file);

  saveAs("Measurements", DAPI_Measurement_output_plus20 + File.separator  +
"Results.csv");

}

selectWindow("Results");

run("Close");

//-------------------------------------------------------------------------------------------------
DAPI Evaluation: +40 MGI

setBatchMode(true);

processFolder23(DAPI_Evaluation_input);

// function to scan folders/subfolders/files to find files with correct suffix
```

```
function processFolder23 (DAPI_Evaluation_input) {
  list = getFileList(DAPI_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
              processFolder23(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile23(DAPI_Evaluation_input, list[i]);
  }
}
function processFile23 (DAPI_Evaluation_input, file) {
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 174);                  // adjusted brightness and contrast by +20
run("Apply LUT");
//run("Threshold...");
```

```
setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_plus40 + File.separator + file);
  saveAs("Measurements", DAPI_Measurement_output_plus40 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------
DAPI Evaluation: +60 MGI
setBatchMode(true);
processFolder24(DAPI_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder24 (DAPI_Evaluation_input) {
  list = getFileList(DAPI_Evaluation_input);
```

```
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
              processFolder24(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile24(DAPI_Evaluation_input, list[i]);
  }
}
function processFile24 (DAPI_Evaluation_input, file) {
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 134);                  // adjusted brightness and contrast by +30
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                  //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
```

```
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_plus60 + File.separator + file);
  saveAs("Measurements", DAPI_Measurement_output_plus60 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------
DAPI Evaluation: +80 MGI
setBatchMode(true);
processFolder25(DAPI_Evaluation_input);


// function to scan folders/subfolders/files to find files with correct suffix
function processFolder25 (DAPI_Evaluation_input) {
  list = getFileList(DAPI_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
```

```
                processFolder25(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
                processFile25(DAPI_Evaluation_input, list[i]);
  }
}
function processFile25 (DAPI_Evaluation_input, file) {
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 95);                    // adjusted brightness and contrast by +40
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                    //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_plus80 + File.separator + file);
```

```
  saveAs("Measurements", DAPI_Measurement_output_plus80 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------------
DAPI Evaluation: +100 MGI
setBatchMode(true);
processFolder26(DAPI_Evaluation_input);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder26 (DAPI_Evaluation_input) {
  list = getFileList(DAPI_Evaluation_input);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))
              processFolder26(DAPI_Evaluation_input + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile26(DAPI_Evaluation_input, list[i]);
```

```
  }
}
function processFile26 (DAPI_Evaluation_input, file) {
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 55);                    // adjusted brightness and contrast by +50
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                    //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_plus100 + File.separator + file);
  saveAs("Measurements", DAPI_Measurement_output_plus100 + File.separator  +
"Results.csv");
}
selectWindow("Results");
```

```
run("Close");

//-------------------------------------------------------------------------------------------------

DAPI Evaluation: +120 MGI

setBatchMode(true);

processFolder27(DAPI_Evaluation_input);



// function to scan folders/subfolders/files to find files with correct suffix

function processFolder27 (DAPI_Evaluation_input) {

  list = getFileList(DAPI_Evaluation_input);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(DAPI_Evaluation_input + File.separator + list[i]))

              processFolder27(DAPI_Evaluation_input + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile27(DAPI_Evaluation_input, list[i]);

  }

}

function processFile27 (DAPI_Evaluation_input, file) {
```

```
  open(DAPI_Evaluation_input + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(0, 15);                    // adjusted brightness and contrast by +60
run("Apply LUT");
//run("Threshold...");
setThreshold(0, 30);
                                                    //  <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
  saveAs("Tiff", DAPI_Measurement_output_plus120 + File.separator + file);
  saveAs("Measurements", DAPI_Measurement_output_plus120 + File.separator  +
"Results.csv");
}
selectWindow("Results");
run("Close");
//-------------------------------------------------------------------------------------------------------
CLOSE
```

```
beep();

Dialog.create("Finished");

Dialog.addMessage("          All Images Have Been Successfully Converted\n"

+"

                                    \n"

+"

                                       \n"

+"                                  ____                \n"

+"                                  |    |              \n"

+"                                  ====                \n"

+"                                  |    |__            \n"

+"                                  |    |-. \\ \\       \n"

+"                                  |__|    \\ \\        \n"

+"                                   | |        ||     \n"

+"                                  ====  __|            \n"

+"                               _________ ||__              \n"

+"                             /__ Miller  Lab __\\           \n"

+"                                                \n"
```

```
+"
                                        \n"
+"
                                    \n"
+"          MGI Evaluation for NueN and DAPI is complete          \n"


+"
                                    ");
Dialog.show();
```

Appendix C – IC-DRG Script

IC-DRG Script version 1.41

File: IC-DRGs_v141.ijm

Github link: https://github.com/MichaelBAnderson/IC-DRGs_v1.41_FIJI_script

---

This script is operational under ImageJ/FIJI (version v1.53f +)

script begins here:

```
“//
                                        ________________________

//                  Image Cytometry in rat dorsal root ganglia (IC-DRGs) /

//                                                   Version 1.41


//              _________________________  ___________________________/

//                    /                                       \/

//                   /                                          |

//___________/                                      |

//                                     __...--|--...__
```

```
//                                    .-' -=;: :;::`;:::`-.
//                                  .-'"; `-;  :"";`-;   ,;;'`-.
//                                 ."";;:  ;.  ,..   `.;;. .';;'.
//                                / ,;;::,  '; `--'  ,.;: ::  .;:\
//                                    ..- .; ;:,;,;,.,_ _ _ , ; :::;;:..;:.
//                                ;    ;   .;:  /000000\ ::::::::::;., ;
//                             |;. :  ;;;;;;|000MBA00|- ;:::::-"""  |
//                                 | ;,'   ,.:::;;|00201800|  `;::  `;:::|
//                                 ;`-;.   ,;: ,\0000000/." ",     ;;, ;
//                                 '. ."".,  ;;;; _""_ ;;;,;:;  '.,;::.
//                                  `.     ;::, .. .`--'  ,.;: ::   .'
//                                    `.   ' ....   ;-.. . " '    .'
//                                  `-.     ,.,"","";;    .-'
//                                 ""--..._____...--""
//
//                                    Compatible with ImageJ/FIJI version 1.52n
//                 If you are new to FIJI scripts, watch a video tutorial for basic operation:
```

```
//          https://github.com/MichaelBAnderson/IC-DRGs_Tutorials
//                Typical Workflow for Image Cytometry
//                1. Process DAPI and NeuN images with IC-DRGs to generate binary
masks of
//                   neuronal nuclei and soma
//                2. Open CellProfiler software and the included IC-DRGs CellProfiler
template
//                   file
//                3. Copy the NeuN, neuronal nuclei masks, neuronal soma masks, and up
to two
//                   other channels to measure other proteins of interest, into CellProfiler
and Run.
//                4. Upon completion, open the included Microsoft Excel template file and
then
//                copy data from CellProfiler exported data into the template file for
organization.
//
//                                                                    Press "Run"
//-------------------------------------------------------------------------------------\\
```

```
//-------------------------------------------------------------------------------------\\

//                          Script Parameters

//

#@ File (label = "Select folder input - NeuN", style = "directory") NeuN_main

#@ File (label = "Select folder input - DAPI", style = "directory") DAPI_main

#@ File (label = "Select folder output - CellProfiler Neuronal Nuclei ", style =
"directory") CellP_Neuronal_Nuclei

#@ File (label = "Select folder output - CellProfiler Neuronal Soma ", style =
"directory") CellP_Neuronal_Soma

#@ File (label = "Select folder - EMGI Score", style = "directory") EMGI_Results

#@ File (label = "Select folder 01 - Soma 1", style = "directory") Soma_t01

#@ File (label = "Select folder 02 - Soma 2", style = "directory") Soma_t02

#@ File (label = "Select folder 03 - Soma 3", style = "directory") Soma_t03

#@ File (label = "Select folder 04 - Soma 4", style = "directory") Soma_t04

#@ File (label = "Select folder 05 - Soma 5", style = "directory") Soma_t05

#@ File (label = "Select folder 06 - Soma merge 1", style = "directory") Soma_merge_01

#@ File (label = "Select folder 07 - Soma merge 2", style = "directory") Soma_merge_02

#@ File (label = "Select folder 08 - Soma merge 3", style = "directory") Soma_merge_03
```

#@ File (label = "Select folder 09 - Soma merge 4", style = "directory") Soma_merge_04

#@ File (label = "Select folder 10 - Nuclear Process", style = "directory") Nuc_process

#@ File (label = "Select folder 11 - Nuclear Holes", style = "directory") Nuc_holes

#@ File (label = "Select folder 12 - Nuclear 1", style = "directory") Nuc_t01

#@ File (label = "Select folder 13 - Nuclear 2", style = "directory") Nuc_t02

#@ File (label = "Select folder 14 - Nuclear 3", style = "directory") Nuc_t03

#@ File (label = "Select folder 15 - Nuclear 4", style = "directory") Nuc_t04

#@ File (label = "Select folder 16 - Nuclear 5", style = "directory") Nuc_t05

#@ File (label = "Select folder 17 - Nuclear 6", style = "directory") Nuc_t06

#@ File (label = "Select folder 18 - Nuclear 7", style = "directory") Nuc_t07

#@ File (label = "Select folder 19 - Nuclear 8", style = "directory") Nuc_t08

#@ File (label = "Select folder 20 - Nuclear 9", style = "directory") Nuc_t09

#@ File (label = "Select folder 21 - Nuclear 10", style = "directory") Nuc_t10

#@ File (label = "Select folder 22 - Nuclear 11", style = "directory") Nuc_t11

#@ File (label = "Select folder 23 - Nuclear 12", style = "directory") Nuc_t12

#@ File (label = "Select folder 24 - Nuclear merge 1", style = "directory")
Nuc_merge_01

#@ File (label = "Select folder 25 - Nuclear merge 2", style = "directory")

Nuc_merge_02

#@ File (label = "Select folder 26 - Nuclear merge 3", style = "directory")

Nuc_merge_03

#@ File (label = "Select folder 27 - Nuclear merge 4", style = "directory")

Nuc_merge_04

#@ File (label = "Select folder 28 - Nuclear merge 5", style = "directory")

Nuc_merge_05

#@ File (label = "Select folder 29 - Nuclear merge 6", style = "directory")

Nuc_merge_06

#@ File (label = "Select folder 30 - Nuclear merge 7", style = "directory")

Nuc_merge_07

#@ File (label = "Select folder 31 - Nuclear merge 8", style = "directory")

Nuc_merge_08

#@ File (label = "Select folder 32 - Nuclear merge 9", style = "directory")

Nuc_merge_09

#@ File (label = "Select folder 33 - Nuclear merge 10", style = "directory")

Nuc_merge_10

#@ File (label = "Select folder 34 - Nuclear rename 1", style = "directory")

Nuclei_rename_1

```
#@ File (label = "Select folder 35 - Soma rename 1", style = "directory")
Soma_rename_1
#@ String (label = "File suffix", value = ".tif") suffix
//-------------------------------------------------------------------------------Soma 1 Threshold [65] /
```

Macro 01

```
setBatchMode(true);
processFolder54(NeuN_main);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder54 (NeuN_main) {
  list = getFileList(NeuN_main);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(NeuN_main + File.separator + list[i]))
                processFolder54(NeuN_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
                processFile54(NeuN_main, Soma_t01, list[i]);
  }
}
```

```
function processFile54 (NeuN_main, Soma_t01, file) {
   open(NeuN_main + File.separator + file);
setThreshold(65, 255);
run("Convert to Mask");
run("Fill Holes");
run("Analyze Particles...", "  show=Masks exclude");
run("Adjustable Watershed", "tolerance=1.5");
run("Analyze Particles...", "size=700-15000 circularity=0.20-1.00 show=Masks exclude
clear in_situ");
   saveAs("Tiff", Soma_t01 + File.separator + file);
   print("Processing: " + NeuN_main + File.separator + file);
   print("Saving to: " + Soma_t01);
}
close("*");
//------------------------------------------------------------------------Soma 2 Threshold  [70] /
```

Macro 02

```
setBatchMode(true);
processFolder1(NeuN_main);
```

```
// function to scan folders/subfolders/files to find files with correct suffix

function processFolder1 (NeuN_main) {

  list = getFileList(NeuN_main);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_main + File.separator + list[i]))

              processFolder1(NeuN_main + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile1(NeuN_main, Soma_t02, list[i]);

  }

}

function processFile1 (NeuN_main, Soma_t02, file) {

  open(NeuN_main + File.separator + file);

setThreshold(70, 255);

run("Convert to Mask");

run("Fill Holes");

run("Analyze Particles...", "  show=Masks exclude");
```

```
run("Adjustable Watershed", "tolerance=1.5");

run("Analyze Particles...", "size=700-15000 circularity=0.20-1.00 show=Masks exclude
clear in_situ");

  saveAs("Tiff", Soma_t02 + File.separator + file);

  print("Processing: " + NeuN_main + File.separator + file);

  print("Saving to: " + Soma_t02);

}

close("*");

//------------------------------------------------------------------------------Soma 3 Threshold [80] /
```

Macro 03

```
setBatchMode(true);

processFolder2(NeuN_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder2 (NeuN_main) {

  list = getFileList(NeuN_main);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_main + File.separator + list[i]))
```

```
                processFolder2(NeuN_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
                processFile2(NeuN_main, Soma_t03, list[i]);
  }
}
function processFile2 (NeuN_main, Soma_t03, file) {
  open(NeuN_main + File.separator + file);


setThreshold(80, 255);
run("Convert to Mask");
run("Fill Holes");
run("Analyze Particles...", "  show=Masks exclude");
run("Adjustable Watershed", "tolerance=1.5");
run("Analyze Particles...", "size=700-15000 circularity=0.20-1.00 show=Masks exclude
clear in_situ");
  saveAs("Tiff", Soma_t03 + File.separator + file);
  print("Processing: " + NeuN_main + File.separator + file);
  print("Saving to: " + Soma_t03);
```

```
}

close("*");

//--------------------------------------------------------------------------Soma 4 Threshold [110]
```

/ **Macro 04**

```
setBatchMode(true);

processFolder44(NeuN_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder44 (NeuN_main) {

  list = getFileList(NeuN_main);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_main + File.separator + list[i]))

              processFolder44(NeuN_main + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile44(NeuN_main, Soma_t04, list[i]);

  }

}

function processFile44 (NeuN_main, Soma_t04, file) {
```

```
  open(NeuN_main + File.separator + file);
setAutoThreshold("Default");
//run("Threshold...");
setThreshold(110, 255);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Analyze Particles...", "  show=Masks exclude");
run("Analyze Particles...", "size=700-max circularity=0.20-1.00 show=Masks clear
in_situ");
run("Erode");
run("Erode");
run("Fill Holes");
run("Adjustable Watershed", "tolerance=1.5");
  saveAs("Tiff", Soma_t04 + File.separator + file);
  print("Processing: " + NeuN_main + File.separator + file);
  print("Saving to: " + Soma_t04);
}
close("*");
```

//----------Soma 5 Threshold [140] /

Macro 05

```
setBatchMode(true);

processFolder46(NeuN_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder46 (NeuN_main) {

  list = getFileList(NeuN_main);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(NeuN_main + File.separator + list[i]))

              processFolder46(NeuN_main + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile46(NeuN_main, Soma_t05, list[i]);

  }

}

function processFile46 (NeuN_main, Soma_t05, file) {

  open(NeuN_main + File.separator + file);

setAutoThreshold("Default");
```

```
//run("Threshold...");

setThreshold(140, 255);

setOption("BlackBackground", false);

run("Convert to Mask");

run("Analyze Particles...", "  show=Masks exclude");

run("Analyze Particles...", "size=700-max circularity=0.20-1.00 show=Masks clear
in_situ");

run("Erode");

run("Erode");

run("Fill Holes");

run("Adjustable Watershed", "tolerance=1.5");

  saveAs("Tiff", Soma_t05 + File.separator + file);

  print("Processing: " + NeuN_main + File.separator + file);

  print("Saving to: " + Soma_t05);

}

close("*");

//--------------------------------------------------------------------------folder 51: NeuN Merge 1
```
/ **Macro 06**

```
setBatchMode(true);

processFolder45(Soma_t02, Soma_t03);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder45(Soma_t02, Soma_t03) {

	list1 = getFileList(Soma_t02);

	list1 = Array.sort(list1);

	list2 = getFileList(Soma_t03);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Soma_t02 + File.separator + list1[i]) &&
File.isDirectory(Soma_t03 + File.separator + list2[i]))

				processFolder45(Soma_t02 + File.separator + list1[i],
Soma_t02 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))

				processFile45(Soma_t02, Soma_t03, Soma_merge_01,
list1[i], list2[i]);

		}
```

```
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile45(Soma_t02, Soma_t03, Soma_merge_01, file1, file2) {
  open(Soma_t02 + File.separator + file1);
  id1 = getImageID();
  open(Soma_t03 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Add ", id2, id1);

        saveAs("Tiff", Soma_merge_01 + File.separator + file1);
        print("Processing: " + Soma_t02 + File.separator + file1);
        print("Processing: " + Soma_t03 + File.separator + file2);
        print("Saving to: " + Soma_merge_01);
}
close("*");
```

```
//-------------------------------------------------------------------------folder 53: NeuN Merge 2 / Macro 07

setBatchMode(true);

processFolder48(Soma_t04, Soma_t05);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder48(Soma_t04, Soma_t05) {

	list1 = getFileList(Soma_t04);

	list1 = Array.sort(list1);

	list2 = getFileList(Soma_t05);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Soma_t04 + File.separator + list1[i]) && File.isDirectory(Soma_t05 + File.separator + list2[i]))

				processFolder48(Soma_t04 + File.separator + list1[i], Soma_t04 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                                processFile48(Soma_t04, Soma_t05, Soma_merge_02,
list1[i], list2[i]);
                        }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile48(Soma_t04, Soma_t05, Soma_merge_02, file1, file2) {
  open(Soma_t04 + File.separator + file1);
  id1 = getImageID();
  open(Soma_t05 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Add ", id2, id1);
        saveAs("Tiff", Soma_merge_02 + File.separator + file1);
        print("Processing: " + Soma_t04 + File.separator + file1);
        print("Processing: " + Soma_t05 + File.separator + file2);
        print("Saving to: " + Soma_merge_02);
}
```

```
close("*");

//-------------------------------------------------------------------------folder 4: NeuN Merge
```
3 / **Macro 08**

```
setBatchMode(true);

processFolder50(Soma_merge_01, Soma_merge_02);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder50(Soma_merge_01, Soma_merge_02) {

	list1 = getFileList(Soma_merge_01);

	list1 = Array.sort(list1);

	list2 = getFileList(Soma_merge_02);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Soma_merge_01 + File.separator + list1[i]) &&
File.isDirectory(Soma_t03 + File.separator + list2[i]))

				processFolder50(Soma_merge_01 + File.separator +
list1[i], Soma_merge_01 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                                processFile50(Soma_merge_01, Soma_merge_02,
Soma_merge_03, list1[i], list2[i]);
                        }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile50(Soma_merge_01, Soma_merge_02, Soma_merge_03, file1, file2)
{
 open(Soma_merge_01 + File.separator + file1);
 id1 = getImageID();
 open(Soma_merge_02 + File.separator + file2);
 id2 = getImageID();
 imageCalculator("Add ", id2, id1);
 run("Adjustable Watershed", "tolerance=1.5");
        saveAs("Tiff", Soma_merge_03 + File.separator + file1);
        print("Processing: " + Soma_merge_01 + File.separator + file1);
        print("Processing: " + Soma_merge_02 + File.separator + file2);
```

```
		print("Saving to: " + Soma_merge_03);
}
close("*");
//-----------------------------------------------------folder 4: NeuN Merge 3: Neuronal Mask
```
/ **Macro 09**

```
setBatchMode(true);
processFolder49(Soma_t01, Soma_merge_03);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder49(Soma_t01, Soma_merge_03) {
		list1 = getFileList(Soma_t01);
		list1 = Array.sort(list1);
		list2 = getFileList(Soma_merge_03);
		list2 = Array.sort(list2);
		if (list1.length == list2.length) {
			for (i = 0; i < list1.length; i++) {
				if(File.isDirectory(Soma_t01 + File.separator + list1[i]) &&
File.isDirectory(Soma_t03 + File.separator + list2[i]))
```

```
                                processFolder49(Soma_t01 + File.separator + list1[i],
Soma_t01 + File.separator + list1[i]);
                            if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
                                processFile49(Soma_t01, Soma_merge_03,
Soma_merge_04, list1[i], list2[i]);
                    }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile49(Soma_t01, Soma_merge_03, Soma_merge_04, file1, file2) {
  open(Soma_t01 + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_03 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Add ", id2, id1);
  run("Adjustable Watershed", "tolerance=1.5");
        saveAs("Tiff", Soma_merge_04 + File.separator + file1);
```

```
        print("Processing: " + Soma_t01 + File.separator + file1);
        print("Processing: " + Soma_merge_03 + File.separator + file2);
        print("Saving to: " + Soma_merge_04);
}
selectWindow("Log");
run("Close");
//------------------------------------------------------------------------------Nuclear 1: DAPI 60/125
```
/ **Macro 10**
```
//------------------------------------------------------------------ Folder 5-6: DAPI in / DAPI out
setBatchMode(true);
processFolder51(DAPI_main);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder51 (DAPI_main) {
  list = getFileList(DAPI_main);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
                processFolder51(DAPI_main + File.separator + list[i]);
```

```
        if(endsWith(list[i], suffix))
                processFile51(DAPI_main, Nuc_process, list[i]);
  }
}
function processFile51 (DAPI_main, Nuc_process, file) {
  open(DAPI_main + File.separator + file);
//run("Threshold...");
setThreshold(60, 255);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Analyze Particles...", "size=500-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");
run("Fill Holes");
 run("Erode");
 run("Erode");
  saveAs("Tiff", Nuc_process + File.separator + file);
  print("Processing: " + DAPI_main + File.separator + file);
  print("Saving to: " + Nuc_process);
```

```
}

close("*");

//-------------------------------------------------------------------------------------------------
```

-- / **Macro 11**

```
setBatchMode(true);

processFolder52(Nuc_process, Soma_merge_04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder52(Nuc_process, Soma_merge_04) {

	list1 = getFileList(Nuc_process);

	list1 = Array.sort(list1);

	list2 = getFileList(Soma_merge_04);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))

				processFolder52(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);
```

```
                        if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
                                processFile52(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
                    }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile52(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
```

```
}

close("*");

//-------------------------------------------------------------------------------------------------
```

-- / **Macro 12**

```
setBatchMode(true);

processFolder53(Nuc_holes);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder53 (Nuc_holes) {

  list = getFileList(Nuc_holes);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(Nuc_holes + File.separator + list[i]))

              processFolder53(Nuc_holes + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile53(Nuc_holes, Nuc_t01, list[i]);

  }

}

function processFile53 (Nuc_holes, Nuc_t01, file) {
```

```
   open(Nuc_holes + File.separator + file);



 //run("Threshold...");

//setThreshold(0, 0);

setOption("BlackBackground", false);

run("Convert to Mask");

run("Invert");

run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");

run("Invert");

   saveAs("Tiff", Nuc_t01 + File.separator + file);

   print("Processing: " + Nuc_holes + File.separator + file);

   print("Saving to: " + Nuc_t01);

}

close("*");
```

//----------Nuclear 2: DAPI 68/255 / **Macro 13**

//---------- Folder 5-6: DAPI in / DAPI out

```
setBatchMode(true);
processFolder70(DAPI_main);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder70 (DAPI_main) {
  list = getFileList(DAPI_main);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
              processFolder70(DAPI_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile70(DAPI_main, Nuc_process, list[i]);
  }
}
function processFile70 (DAPI_main, Nuc_process, file) {
  open(DAPI_main + File.separator + file);
//run("Threshold...");
setThreshold(68, 255);
```

```
setOption("BlackBackground", false);

run("Convert to Mask");

run("Analyze Particles...", "size=500-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");

run("Fill Holes");

 run("Erode");

 run("Erode");

  saveAs("Tiff", Nuc_process + File.separator + file);

  print("Processing: " + DAPI_main + File.separator + file);

  print("Saving to: " + Nuc_process);

}

close("*");

//-------------------------------------------------------------------------------------------------
```

-- / **Macro 14**

```
setBatchMode(true);

processFolder55(Nuc_process, Soma_merge_04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder55(Nuc_process, Soma_merge_04) {
```

```
	list1 = getFileList(Nuc_process);
	list1 = Array.sort(list1);
	list2 = getFileList(Soma_merge_04);
	list2 = Array.sort(list2);
	if (list1.length == list2.length) {
		for (i = 0; i < list1.length; i++) {
			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))
				processFolder55(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);
			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
				processFile55(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
		}
	} else {
	print("The two directories do not have the same # of files!");
}
}
```

```
function processFile55(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
}
close("*");
//-------------------------------------------------------------------------------------------------
```

-- / **Macro 15**

```
setBatchMode(true);
processFolder56(Nuc_holes);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder56 (Nuc_holes) {
```

```
  list = getFileList(Nuc_holes);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Nuc_holes + File.separator + list[i]))
              processFolder56(Nuc_holes + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile56(Nuc_holes, Nuc_t02, list[i]);
  }
}
function processFile56 (Nuc_holes, Nuc_t02, file) {
  open(Nuc_holes + File.separator + file);
 //run("Threshold...");
//setThreshold(0, 0);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Invert");
run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");
```

```
run("Invert");
  saveAs("Tiff", Nuc_t02 + File.separator + file);
  print("Processing: " + Nuc_holes + File.separator + file);
  print("Saving to: " + Nuc_t02);
}
close("*");
```

//----------Nuclear 3: DAPI 75/255 / **Macro 16**

```
//---------- Folder 5-6: DAPI in / DAPI out
setBatchMode(true);
processFolder4(DAPI_main);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder4 (DAPI_main) {
  list = getFileList(DAPI_main);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
                processFolder4(DAPI_main + File.separator + list[i]);
```

```
        if(endsWith(list[i], suffix))
                processFile4(DAPI_main, Nuc_process, list[i]);
  }
}
function processFile4 (DAPI_main, Nuc_process, file) {
  open(DAPI_main + File.separator + file);
//run("Threshold...");
setThreshold(75, 255);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Analyze Particles...", "size=500-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");
run("Fill Holes");
 run("Erode");
 run("Erode");
  saveAs("Tiff", Nuc_process + File.separator + file);
  print("Processing: " + DAPI_main + File.separator + file);
  print("Saving to: " + Nuc_process);
```

```
}

close("*");

//-------------------------------------------------------------------------------------------------
```

-- / **Macro 17**

```
setBatchMode(true);

processFolder5(Nuc_process, Soma_merge_04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder5(Nuc_process, Soma_merge_04) {

	list1 = getFileList(Nuc_process);

	list1 = Array.sort(list1);

	list2 = getFileList(Soma_merge_04);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))

				processFolder5(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);
```

```
                        if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
                                processFile5(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
                    }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile5(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
```

```
}

close("*");

//-------------------------------------------------------------------------------------------------
```

-- / **Macro 18**

```
setBatchMode(true);

processFolder6(Nuc_holes);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder6 (Nuc_holes) {

  list = getFileList(Nuc_holes);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(Nuc_holes + File.separator + list[i]))

              processFolder6(Nuc_holes + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile6(Nuc_holes, Nuc_t03, list[i]);

  }

}

function processFile6 (Nuc_holes, Nuc_t03, file) {
```

```
   open(Nuc_holes + File.separator + file);

 //run("Threshold...");

//setThreshold(0, 0);

setOption("BlackBackground", false);

run("Convert to Mask");

run("Invert");

run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");

run("Invert");

   saveAs("Tiff", Nuc_t03 + File.separator + file);

   print("Processing: " + Nuc_holes + File.separator + file);

   print("Saving to: " + Nuc_t03);

}

close("*");

//--------------------------------------------------------------------------Nuclear 4: DAPI
```

85/255 / **Macro 19**

```
setBatchMode(true);

processFolder7(DAPI_main);
```

```
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder7 (DAPI_main) {
  list = getFileList(DAPI_main);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
              processFolder7(DAPI_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile7(DAPI_main, Nuc_process, list[i]);
  }
}
function processFile7 (DAPI_main, Nuc_process, file) {
  open(DAPI_main + File.separator + file);
//run("Threshold...");
setThreshold(85, 255);
setOption("BlackBackground", false);
run("Convert to Mask");
```

```
run("Analyze Particles...", "size=500-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");

run("Fill Holes");

run("Erode");

run("Erode");

  saveAs("Tiff", Nuc_process + File.separator + file);

  print("Processing: " + DAPI_main + File.separator + file);

  print("Saving to: " + Nuc_process);

}

close("*");

//-------------------------------------------------------------------------------------------------------
```
--- / **Macro 20**

```
setBatchMode(true);

processFolder8(Nuc_process, Soma_merge_04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder8(Nuc_process, Soma_merge_04) {

        list1 = getFileList(Nuc_process);

        list1 = Array.sort(list1);
```

```
	list2 = getFileList(Soma_merge_04);
	list2 = Array.sort(list2);
	if (list1.length == list2.length) {
		for (i = 0; i < list1.length; i++) {
			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))
				processFolder8(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);
			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
				processFile8(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
		}
	} else {
	print("The two directories do not have the same # of files!");
}
}
function processFile8(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
```

```
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
}
close("*");
//-------------------------------------------------------------------------------------------------
```
--- / **Macro 21**

```
setBatchMode(true);
processFolder9(Nuc_holes);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder9 (Nuc_holes) {
  list = getFileList(Nuc_holes);
  list = Array.sort(list);
```

```
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Nuc_holes + File.separator + list[i]))
            processFolder9(Nuc_holes + File.separator + list[i]);
        if(endsWith(list[i], suffix))
            processFile9(Nuc_holes, Nuc_t04, list[i]);
  }
}
function processFile9 (Nuc_holes, Nuc_t04, file) {
  open(Nuc_holes + File.separator + file);
 //run("Threshold...");
//setThreshold(0, 0);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Invert");
run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");
run("Invert");
  saveAs("Tiff", Nuc_t04 + File.separator + file);
```

```
  print("Processing: " + Nuc_holes + File.separator + file);
  print("Saving to: " + Nuc_t04);
}
close("*");
//-------------------------------------------------------------------------------Nuclear 5: DAPI 95/255
```
/ **Macro 22**

```
setBatchMode(true);
processFolder10(DAPI_main);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder10 (DAPI_main) {
  list = getFileList(DAPI_main);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
              processFolder10(DAPI_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile10(DAPI_main, Nuc_process, list[i]);
  }
```

```
}

function processFile10 (DAPI_main, Nuc_process, file) {

  open(DAPI_main + File.separator + file);

//run("Threshold...");

setThreshold(95, 255);

setOption("BlackBackground", false);

run("Convert to Mask");

run("Analyze Particles...", "size=500-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");

run("Fill Holes");

run("Erode");

run("Erode");

  saveAs("Tiff", Nuc_process + File.separator + file);

  print("Processing: " + DAPI_main + File.separator + file);

  print("Saving to: " + Nuc_process);

}

close("*");
```

```
//-------------------------------------------------------------------------------------------------------
```

--- / **Macro 23**

```
setBatchMode(true);

processFolder11(Nuc_process, Soma_merge_04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder11(Nuc_process, Soma_merge_04) {

	list1 = getFileList(Nuc_process);

	list1 = Array.sort(list1);

	list2 = getFileList(Soma_merge_04);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))

				processFolder11(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                                processFile11(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
                        }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile11(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
}
```

```
close("*");

//-------------------------------------------------------------------------------------------------
```

--- / **Macro 24**

```
setBatchMode(true);

processFolder12(Nuc_holes);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder12 (Nuc_holes) {

  list = getFileList(Nuc_holes);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(Nuc_holes + File.separator + list[i]))

              processFolder12(Nuc_holes + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile12(Nuc_holes, Nuc_t05, list[i]);

  }

}

function processFile12 (Nuc_holes, Nuc_t05, file) {

  open(Nuc_holes + File.separator + file);
```

```
 //run("Threshold...");

//setThreshold(0, 0);

setOption("BlackBackground", false);

run("Convert to Mask");

run("Invert");

run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");

run("Invert");

   saveAs("Tiff", Nuc_t05 + File.separator + file);

   print("Processing: " + Nuc_holes + File.separator + file);

   print("Saving to: " + Nuc_t05);

}

close("*");

//------------------------------------------------------------------------Nuclear 6: DAPI 115/180 /
```

Macro 25

```
setBatchMode(true);

processFolder13(DAPI_main);

// function to scan folders/subfolders/files to find files with correct suffix
```

```
function processFolder13 (DAPI_main) {
  list = getFileList(DAPI_main);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
              processFolder13(DAPI_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile13(DAPI_main, Nuc_process, list[i]);
  }
}
function processFile13 (DAPI_main, Nuc_process, file) {
  open(DAPI_main + File.separator + file);
//run("Threshold...");
setThreshold(115, 255);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Analyze Particles...", "size=400-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");
```

```
run("Fill Holes");

run("Erode");

run("Erode");



  saveAs("Tiff", Nuc_process + File.separator + file);

  print("Processing: " + DAPI_main + File.separator + file);

  print("Saving to: " + Nuc_process);

}

close("*");

//-------------------------------------------------------------------------------------------------------
```

-- / **Macro 26**

```
setBatchMode(true);

processFolder14(Nuc_process, Soma_merge_04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder14(Nuc_process, Soma_merge_04) {

        list1 = getFileList(Nuc_process);

        list1 = Array.sort(list1);

        list2 = getFileList(Soma_merge_04);
```

```
	list2 = Array.sort(list2);
	if (list1.length == list2.length) {
		for (i = 0; i < list1.length; i++) {
			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))
				processFolder14(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);
			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
				processFile14(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
		}
	} else {
	print("The two directories do not have the same # of files!");
}
}
function processFile14(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
```

```
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
}
close("*");
//-------------------------------------------------------------------------------------------------
```

-- / **Macro 27**

```
setBatchMode(true);
processFolder15(Nuc_holes);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder15 (Nuc_holes) {
  list = getFileList(Nuc_holes);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
```

```
        if(File.isDirectory(Nuc_holes + File.separator + list[i]))
                processFolder15(Nuc_holes + File.separator + list[i]);
        if(endsWith(list[i], suffix))
                processFile15(Nuc_holes, Nuc_t06, list[i]);
  }
}
function processFile15 (Nuc_holes, Nuc_t06, file) {
  open(Nuc_holes + File.separator + file);
 //run("Threshold...");
//setThreshold(0, 0);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Invert");
run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");
run("Invert");
  saveAs("Tiff", Nuc_t06 + File.separator + file);
  print("Processing: " + Nuc_holes + File.separator + file);
```

```
  print("Saving to: " + Nuc_t06);
}
close("*");
//--------------------------------------------------------------------------Nuclear 7: DAPI 135/255
/ Macro 28
setBatchMode(true);
processFolder16(DAPI_main);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder16 (DAPI_main) {
  list = getFileList(DAPI_main);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
              processFolder16(DAPI_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile16(DAPI_main, Nuc_process, list[i]);
  }
}
```

```
function processFile16 (DAPI_main, Nuc_process, file) {
   open(DAPI_main + File.separator + file);
//run("Threshold...");
setThreshold(135, 255);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Analyze Particles...", "size=400-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");
run("Fill Holes");
run("Erode");
run("Erode");
   saveAs("Tiff", Nuc_process + File.separator + file);
   print("Processing: " + DAPI_main + File.separator + file);
   print("Saving to: " + Nuc_process);
}
close("*");
//-------------------------------------------------------------------------------------------------
```

-- / **Macro 29**

```
setBatchMode(true);
processFolder17(Nuc_process, Soma_merge_04);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder17(Nuc_process, Soma_merge_04) {
	list1 = getFileList(Nuc_process);
	list1 = Array.sort(list1);
	list2 = getFileList(Soma_merge_04);
	list2 = Array.sort(list2);
	if (list1.length == list2.length) {
		for (i = 0; i < list1.length; i++) {
			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))
				processFolder17(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);
			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
				processFile17(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
		}
```

```
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile17(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
}
close("*");
//-------------------------------------------------------------------------------------------------
```

-- / **Macro 30**

```
setBatchMode(true);
processFolder18(Nuc_holes);


// function to scan folders/subfolders/files to find files with correct suffix
function processFolder18 (Nuc_holes) {
  list = getFileList(Nuc_holes);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Nuc_holes + File.separator + list[i]))
                processFolder18(Nuc_holes + File.separator + list[i]);
        if(endsWith(list[i], suffix))
                processFile18(Nuc_holes, Nuc_t07, list[i]);
  }
}
function processFile18 (Nuc_holes, Nuc_t07, file) {
  open(Nuc_holes + File.separator + file);
 //run("Threshold...");
```

```
//setThreshold(0, 0);

setOption("BlackBackground", false);

run("Convert to Mask");

run("Invert");

run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");

run("Invert");

  saveAs("Tiff", Nuc_t07 + File.separator + file);

  print("Processing: " + Nuc_holes + File.separator + file);

  print("Saving to: " + Nuc_t07);

}

close("*");

//--------------------------------------------------------------------------Nuclear 8: DAPI 155/255
```
/ **Macro 31**

```
setBatchMode(true);

processFolder19(DAPI_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder19 (DAPI_main) {
```

```
  list = getFileList(DAPI_main);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
              processFolder19(DAPI_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile19(DAPI_main, Nuc_process, list[i]);
  }
}
function processFile19 (DAPI_main, Nuc_process, file) {
  open(DAPI_main + File.separator + file);
setAutoThreshold("Default");
//run("Threshold...");
setThreshold(115, 190);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Analyze Particles...", "size=200-5000 circularity=0.20-1.00 show=Masks exclude
clear");
```

```
run("Fill Holes");

run("Invert");

  saveAs("Tiff", Nuc_process + File.separator + file);

  print("Processing: " + DAPI_main + File.separator + file);

  print("Saving to: " + Nuc_process);

}

close("*");

//-------------------------------------------------------------------------------------------------------
```

-- / **Macro 32**

```
setBatchMode(true);

processFolder20(Nuc_process, Soma_merge_04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder20(Nuc_process, Soma_merge_04) {

        list1 = getFileList(Nuc_process);

        list1 = Array.sort(list1);

        list2 = getFileList(Soma_merge_04);

        list2 = Array.sort(list2);

        if (list1.length == list2.length) {
```

```
		for (i = 0; i < list1.length; i++) {
			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))
				processFolder20(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);
			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
				processFile20(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
		}
	} else {
	print("The two directories do not have the same # of files!");
}
}
function processFile20(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
```

```
 imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
}
close("*");
//-------------------------------------------------------------------------------------------------
```
-- / **Macro 33**

```
setBatchMode(true);
processFolder21(Nuc_holes);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder21 (Nuc_holes) {
  list = getFileList(Nuc_holes);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Nuc_holes + File.separator + list[i]))
                processFolder21(Nuc_holes + File.separator + list[i]);
```

```
        if(endsWith(list[i], suffix))
                processFile21(Nuc_holes, Nuc_t08, list[i]);
  }
}
function processFile21 (Nuc_holes, Nuc_t08, file) {
  open(Nuc_holes + File.separator + file);
 //run("Threshold...");
//setThreshold(0, 0);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Invert");
run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");
run("Invert");
  saveAs("Tiff", Nuc_t08 + File.separator + file);
  print("Processing: " + Nuc_holes + File.separator + file);
  print("Saving to: " + Nuc_t08);
}
```

```
close("*");



//--------------------------------------------------------------------------Nuclear 9: DAPI
```

175/255 / **Macro 34**

```
setBatchMode(true);

processFolder22(DAPI_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder22 (DAPI_main) {

  list = getFileList(DAPI_main);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(DAPI_main + File.separator + list[i]))

              processFolder22(DAPI_main + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile22(DAPI_main, Nuc_process, list[i]);

  }

}

function processFile22 (DAPI_main, Nuc_process, file) {
```

```
   open(DAPI_main + File.separator + file);

//run("Threshold...");

setThreshold(175, 255);

setOption("BlackBackground", false);

run("Convert to Mask");

run("Analyze Particles...", "size=500-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");

run("Fill Holes");

run("Erode");

run("Erode");

   saveAs("Tiff", Nuc_process + File.separator + file);

   print("Processing: " + DAPI_main + File.separator + file);

   print("Saving to: " + Nuc_process);

}

close("*");

//-------------------------------------------------------------------------------------------------
```

-- / **Macro 35**

```
setBatchMode(true);
```

```
processFolder23(Nuc_process, Soma_merge_04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder23(Nuc_process, Soma_merge_04) {

	list1 = getFileList(Nuc_process);

	list1 = Array.sort(list1);

	list2 = getFileList(Soma_merge_04);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))

				processFolder23(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))

				processFile23(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);

		}

	} else {
```

```
        print("The two directories do not have the same # of files!");
}
}
function processFile23(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
}
close("*");
//-------------------------------------------------------------------------------------------------
```

-- / **Macro 36**

```
setBatchMode(true);
```

```
processFolder24(Nuc_holes);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder24 (Nuc_holes) {

  list = getFileList(Nuc_holes);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(Nuc_holes + File.separator + list[i]))

              processFolder24(Nuc_holes + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile24(Nuc_holes, Nuc_t09, list[i]);

  }

}

function processFile24 (Nuc_holes, Nuc_t09, file) {

  open(Nuc_holes + File.separator + file);

 //run("Threshold...");

//setThreshold(0, 0);

setOption("BlackBackground", false);
```

```
run("Convert to Mask");

run("Invert");

run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");

run("Invert");

  saveAs("Tiff", Nuc_t09 + File.separator + file);

  print("Processing: " + Nuc_holes + File.separator + file);

  print("Saving to: " + Nuc_t09);

}

close("*");



//--------------------------------------------------------------------------Nuclear 10: DAPI
```

195/255 / **Macro 37**

```
setBatchMode(true);

processFolder25(DAPI_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder25 (DAPI_main) {

  list = getFileList(DAPI_main);
```

```
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
              processFolder25(DAPI_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile25(DAPI_main, Nuc_process, list[i]);
  }
}
function processFile25 (DAPI_main, Nuc_process, file) {
  open(DAPI_main + File.separator + file);
//run("Threshold...");
setThreshold(195, 255);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Analyze Particles...", "size=500-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");
run("Fill Holes");
run("Erode");
```

```
run("Erode");
   saveAs("Tiff", Nuc_process + File.separator + file);
   print("Processing: " + DAPI_main + File.separator + file);
   print("Saving to: " + Nuc_process);
}
close("*");


//-------------------------------------------------------------------------------------------------------
```

-- / **Macro 38**

```
setBatchMode(true);
processFolder26(Nuc_process, Soma_merge_04);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder26(Nuc_process, Soma_merge_04) {
        list1 = getFileList(Nuc_process);
        list1 = Array.sort(list1);
        list2 = getFileList(Soma_merge_04);
        list2 = Array.sort(list2);
        if (list1.length == list2.length) {
```

```
		for (i = 0; i < list1.length; i++) {
			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))
				processFolder26(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);
			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
				processFile26(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
		}
	} else {
	print("The two directories do not have the same # of files!");
}
}
function processFile26(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
```

```
 imageCalculator("Subtract ", id2, id1);
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
}
close("*");
//-------------------------------------------------------------------------------------------------
```
-- / **Macro 39**

```
setBatchMode(true);
processFolder27(Nuc_holes);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder27 (Nuc_holes) {
  list = getFileList(Nuc_holes);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Nuc_holes + File.separator + list[i]))
                processFolder27(Nuc_holes + File.separator + list[i]);
```

```
        if(endsWith(list[i], suffix))
                processFile27(Nuc_holes, Nuc_t10, list[i]);
  }
}
function processFile27 (Nuc_holes, Nuc_t10, file) {
  open(Nuc_holes + File.separator + file);
 //run("Threshold...");
//setThreshold(0, 0);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Invert");
run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");
run("Invert");
  saveAs("Tiff", Nuc_t10 + File.separator + file);
  print("Processing: " + Nuc_holes + File.separator + file);
  print("Saving to: " + Nuc_t10);
}
```

```
close("*");

//--------------------------------------------------------------------------Nuclear 11: DAPI 215/255 / Macro 40

setBatchMode(true);

processFolder28(DAPI_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder28 (DAPI_main) {

  list = getFileList(DAPI_main);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(DAPI_main + File.separator + list[i]))

              processFolder28(DAPI_main + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile28(DAPI_main, Nuc_process, list[i]);

  }

}

function processFile28 (DAPI_main, Nuc_process, file) {

  open(DAPI_main + File.separator + file);
```

```
//run("Threshold...");

setThreshold(215, 255);

setOption("BlackBackground", false);

run("Convert to Mask");

run("Analyze Particles...", "size=500-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");

run("Fill Holes");

run("Erode");

run("Erode");

  saveAs("Tiff", Nuc_process + File.separator + file);

  print("Processing: " + DAPI_main + File.separator + file);

  print("Saving to: " + Nuc_process);

}

close("*");

//-------------------------------------------------------------------------------------------------
```

-- / **Macro 41**

```
setBatchMode(true);

processFolder29(Nuc_process, Soma_merge_04);
```

```
// function to scan folders/subfolders/files to find files with correct suffix

function processFolder29(Nuc_process, Soma_merge_04) {

	list1 = getFileList(Nuc_process);

	list1 = Array.sort(list1);

	list2 = getFileList(Soma_merge_04);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))

				processFolder29(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))

				processFile29(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);

		}

	} else {

	print("The two directories do not have the same # of files!");
```

```
}

}

function processFile29(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {

 open(Nuc_process + File.separator + file1);

 id1 = getImageID();

 open(Soma_merge_04 + File.separator + file2);

 id2 = getImageID();

 imageCalculator("Subtract ", id2, id1);

        saveAs("Tiff", Nuc_holes + File.separator + file1);

        print("Processing: " + Nuc_process + File.separator + file1);

        print("Processing: " + Soma_merge_04 + File.separator + file2);

        print("Saving to: " + Nuc_holes);

}

close("*");

//--------------------------------------------------------------------------------------------------
```

-- / **Macro 42**

```
setBatchMode(true);

processFolder30(Nuc_holes);
```

```
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder30 (Nuc_holes) {
  list = getFileList(Nuc_holes);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Nuc_holes + File.separator + list[i]))
              processFolder30(Nuc_holes + File.separator + list[i]);
        if(endsWith(list[i], suffix))
              processFile30(Nuc_holes, Nuc_t11, list[i]);
  }
}
function processFile30 (Nuc_holes, Nuc_t11, file) {
  open(Nuc_holes + File.separator + file);
 //run("Threshold...");
//setThreshold(0, 0);
setOption("BlackBackground", false);
run("Convert to Mask");
```

```
run("Invert");

run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");

run("Invert");

  saveAs("Tiff", Nuc_t11 + File.separator + file);

  print("Processing: " + Nuc_holes + File.separator + file);

  print("Saving to: " + Nuc_t11);

}

close("*");

//------------------------------------------------------------------------Nuclear 12: DAPI
225/255 / Macro 43

//-----------------------------------------------------------------

setBatchMode(true);

processFolder31(DAPI_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder31 (DAPI_main) {

  list = getFileList(DAPI_main);

  list = Array.sort(list);
```

```
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(DAPI_main + File.separator + list[i]))
            processFolder31(DAPI_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
            processFile31(DAPI_main, Nuc_process, list[i]);
  }
}
function processFile31 (DAPI_main, Nuc_process, file) {
  open(DAPI_main + File.separator + file);
//run("Threshold...");
setThreshold(225, 255);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Analyze Particles...", "size=500-5000 circularity=.2-1.00 show=Masks exclude clear
in_situ");
run("Fill Holes");
run("Erode");
run("Erode");
```

```
  saveAs("Tiff", Nuc_process + File.separator + file);
  print("Processing: " + DAPI_main + File.separator + file);
  print("Saving to: " + Nuc_process);
}
close("*");
//-------------------------------------------------------------------------------------------------------
```

-- / **Macro 44**

```
setBatchMode(true);
processFolder32(Nuc_process, Soma_merge_04);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder32(Nuc_process, Soma_merge_04) {
        list1 = getFileList(Nuc_process);
        list1 = Array.sort(list1);
        list2 = getFileList(Soma_merge_04);
        list2 = Array.sort(list2);
        if (list1.length == list2.length) {
                for (i = 0; i < list1.length; i++) {
```

```
                        if(File.isDirectory(Nuc_process + File.separator + list1[i]) &&
File.isDirectory(Soma_merge_04 + File.separator + list2[i]))
                                processFolder32(Nuc_process + File.separator + list1[i],
Nuc_process + File.separator + list1[i]);
                        if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
                                processFile32(Nuc_process, Soma_merge_04, Nuc_holes,
list1[i], list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile32(Nuc_process, Soma_merge_04, Nuc_holes, file1, file2) {
  open(Nuc_process + File.separator + file1);
  id1 = getImageID();
  open(Soma_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("Subtract ", id2, id1);
```

```
        saveAs("Tiff", Nuc_holes + File.separator + file1);
        print("Processing: " + Nuc_process + File.separator + file1);
        print("Processing: " + Soma_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_holes);
}
close("*");
//-------------------------------------------------------------------------------------------------------
-- / Macro 45
setBatchMode(true);
processFolder33(Nuc_holes);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder33 (Nuc_holes) {
  list = getFileList(Nuc_holes);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Nuc_holes + File.separator + list[i]))
                processFolder33(Nuc_holes + File.separator + list[i]);
        if(endsWith(list[i], suffix))
```

```
                processFile33(Nuc_holes, Nuc_t12, list[i]);
    }
}
function processFile33 (Nuc_holes, Nuc_t12, file) {
   open(Nuc_holes + File.separator + file);
 //run("Threshold...");
//setThreshold(0, 0);
setOption("BlackBackground", false);
run("Convert to Mask");
run("Invert");
run("Analyze Particles...", "size=350-5000 circularity=.65-1.00 show=Masks exclude
clear in_situ");
run("Invert");
   saveAs("Tiff", Nuc_t12 + File.separator + file);
   print("Processing: " + Nuc_holes + File.separator + file);
   print("Saving to: " + Nuc_t12);
}
selectWindow("Log");
```

```
run("Close");

//--------------------------------------------------------------------------------Seed Temp 1
```
/ **Macro 46**

```
setBatchMode(true);

processFolder34(Nuc_t03, Nuc_t04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder34(Nuc_t03, Nuc_t04) {

	list1 = getFileList(Nuc_t03);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_t04);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_t03 + File.separator + list1[i]) &&
File.isDirectory(Nuc_t04 + File.separator + list2[i]))

				processFolder34(Nuc_t03 + File.separator + list1[i],
Nuc_t03 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                                processFile34(Nuc_t03, Nuc_t04, Nuc_merge_01, list1[i],
list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile34(Nuc_t03, Nuc_t04, Nuc_merge_01, file1, file2) {
  open(Nuc_t03 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_t04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_01 + File.separator + file1);
        print("Processing: " + Nuc_t03 + File.separator + file1);
        print("Processing: " + Nuc_t04 + File.separator + file2);
        print("Saving to: " + Nuc_merge_01);
}
```

```
close("*");

//--------------------------------------------------------------------------------Seed Temp 2
```
/ **Macro 47**

```
setBatchMode(true);

processFolder35(Nuc_t05, Nuc_t06);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder35(Nuc_t05, Nuc_t06) {

	list1 = getFileList(Nuc_t05);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_t06);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_t05 + File.separator + list1[i]) &&
File.isDirectory(Nuc_t06 + File.separator + list2[i]))

				processFolder35(Nuc_t05 + File.separator + list1[i],
Nuc_t05 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                                processFile35(Nuc_t05, Nuc_t06, Nuc_merge_02, list1[i],
list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile35(Nuc_t05, Nuc_t06, Nuc_merge_02, file1, file2) {
  open(Nuc_t05 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_t06 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_02 + File.separator + file1);
        print("Processing: " + Nuc_t05 + File.separator + file1);
        print("Processing: " + Nuc_t06 + File.separator + file2);
        print("Saving to: " + Nuc_merge_02);
}
```

```
close("*");

//--------------------------------------------------------------------------------Seed Temp 3
```
/ **Macro 48**

```
setBatchMode(true);

processFolder36(Nuc_t07, Nuc_t08);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder36(Nuc_t07, Nuc_t08) {

	list1 = getFileList(Nuc_t07);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_t08);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_t07 + File.separator + list1[i]) &&
File.isDirectory(Nuc_t08 + File.separator + list2[i]))

				processFolder36(Nuc_t07 + File.separator + list1[i],
Nuc_t07 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                                processFile36(Nuc_t07, Nuc_t08, Nuc_merge_03, list1[i],
list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile36(Nuc_t07, Nuc_t08, Nuc_merge_03, file1, file2) {
  open(Nuc_t07 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_t08 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_03 + File.separator + file1);
        print("Processing: " + Nuc_t07 + File.separator + file1);
        print("Processing: " + Nuc_t08 + File.separator + file2);
        print("Saving to: " + Nuc_merge_03);
}
```

```
close("*");

//--------------------------------------------------------------------------------Seed Temp 4 /
```

Macro 49

```
setBatchMode(true);

processFolder37(Nuc_t09, Nuc_t10);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder37(Nuc_t09, Nuc_t10) {

	list1 = getFileList(Nuc_t09);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_t10);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_t09 + File.separator + list1[i]) &&
File.isDirectory(Nuc_t10 + File.separator + list2[i]))

				processFolder37(Nuc_t09 + File.separator + list1[i],
Nuc_t09 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                                processFile37(Nuc_t09, Nuc_t10, Nuc_merge_04, list1[i],
list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile37(Nuc_t09, Nuc_t10, Nuc_merge_04, file1, file2) {
  open(Nuc_t09 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_t10 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_04 + File.separator + file1);
        print("Processing: " + Nuc_t09 + File.separator + file1);
        print("Processing: " + Nuc_t10 + File.separator + file2);
        print("Saving to: " + Nuc_merge_04);
}
```

```
close("*");

//--------------------------------------------------------------------------------Seed Temp 5 /
```

Macro 50

```
setBatchMode(true);

processFolder38(Nuc_t11, Nuc_t12);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder38(Nuc_t11, Nuc_t12) {

	list1 = getFileList(Nuc_t11);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_t12);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_t11 + File.separator + list1[i]) &&
File.isDirectory(Nuc_t12 + File.separator + list2[i]))

				processFolder38(Nuc_t11 + File.separator + list1[i],
Nuc_t11 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                processFile38(Nuc_t11, Nuc_t12, Nuc_merge_05, list1[i],
list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile38(Nuc_t11, Nuc_t12, Nuc_merge_05, file1, file2) {
  open(Nuc_t11 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_t12 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_05 + File.separator + file1);
        print("Processing: " + Nuc_t11 + File.separator + file1);
        print("Processing: " + Nuc_t12 + File.separator + file2);
        print("Saving to: " + Nuc_merge_05);
}
```

```
close("*");

//----------------------------------------------------------------------------------------Seed Temp
```

6 / **Macro 51**

```
setBatchMode(true);

processFolder57(Nuc_t01, Nuc_t02);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder57(Nuc_t01, Nuc_t02) {

	list1 = getFileList(Nuc_t01);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_t02);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_t01 + File.separator + list1[i]) &&
File.isDirectory(Nuc_t02 + File.separator + list2[i]))

				processFolder57(Nuc_t01 + File.separator + list1[i],
Nuc_t01 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                                processFile57(Nuc_t01, Nuc_t02, Nuc_merge_06, list1[i],
list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile57(Nuc_t01, Nuc_t02, Nuc_merge_06, file1, file2) {
  open(Nuc_t01 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_t02 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_06 + File.separator + file1);
        print("Processing: " + Nuc_t01 + File.separator + file1);
        print("Processing: " + Nuc_t02 + File.separator + file2);
        print("Saving to: " + Nuc_merge_06);
}
```

```
close("*");

//--------------------------------------------------------------------------------Final Seed 1 /
```

Macro 52

```
setBatchMode(true);

processFolder39(Nuc_merge_01, Nuc_merge_02);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder39(folde18, Nuc_merge_02) {

	list1 = getFileList(Nuc_merge_01);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_merge_02);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_merge_01 + File.separator + list1[i]) &&
File.isDirectory(Nuc_merge_02 + File.separator + list2[i]))

				processFolder39(Nuc_merge_01 + File.separator + list1[i],
Nuc_merge_01 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                processFile39(Nuc_merge_01, Nuc_merge_02,
Nuc_merge_07, list1[i], list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile39(Nuc_merge_01, Nuc_merge_02, Nuc_merge_07, file1, file2) {
  open(Nuc_merge_01 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_merge_02 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_07 + File.separator + file1);
        print("Processing: " + Nuc_merge_01 + File.separator + file1);
        print("Processing: " + Nuc_merge_02 + File.separator + file2);
        print("Saving to: " + Nuc_merge_07);
}
```

```
close("*");

//----------------------------------------------------------------------------------Final Seed 2
```

/ **Macro 53**

```
setBatchMode(true);

processFolder40(Nuc_merge_03, Nuc_merge_04);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder40(Nuc_merge_03, Nuc_merge_04) {

	list1 = getFileList(Nuc_merge_03);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_merge_04);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_merge_03 + File.separator + list1[i]) &&
File.isDirectory(Nuc_merge_04 + File.separator + list2[i]))

				processFolder40(Nuc_merge_03 + File.separator + list1[i],
Nuc_merge_03 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                    processFile40(Nuc_merge_03, Nuc_merge_04,
Nuc_merge_08, list1[i], list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile40(Nuc_merge_03, Nuc_merge_04, Nuc_merge_08, file1, file2) {
  open(Nuc_merge_03 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_merge_04 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_08 + File.separator + file1);
        print("Processing: " + Nuc_merge_03 + File.separator + file1);
        print("Processing: " + Nuc_merge_04 + File.separator + file2);
        print("Saving to: " + Nuc_merge_08);
}
```

```
close("*");

//--------------------------------------------------------------------------------Final Seed 3 /
```

Macro 54

```
setBatchMode(true);

processFolder41(Nuc_merge_05, Nuc_merge_06);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder41(Nuc_merge_05, Nuc_merge_06) {

	list1 = getFileList(Nuc_merge_05);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_merge_06);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_merge_05 + File.separator + list1[i]) &&
File.isDirectory(Nuc_merge_06 + File.separator + list2[i]))

				processFolder41(Nuc_merge_05 + File.separator + list1[i],
Nuc_merge_05 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                    processFile41(Nuc_merge_05, Nuc_merge_06,
Nuc_merge_09, list1[i], list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile41(Nuc_merge_05, Nuc_merge_06, Nuc_merge_09, file1, file2) {
  open(Nuc_merge_05 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_merge_06 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_09 + File.separator + file1);
        print("Processing: " + Nuc_merge_05 + File.separator + file1);
        print("Processing: " + Nuc_merge_06 + File.separator + file2);
        print("Saving to: " + Nuc_merge_09);
}
```

```
close("*");

//--------------------------------------------------------------------------------Final Seed 4
```
/ **Macro 55**

```
setBatchMode(true);

processFolder59(Nuc_merge_07, Nuc_merge_08);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder59(Nuc_merge_07, Nuc_merge_08) {

	list1 = getFileList(Nuc_merge_07);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_merge_08);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_merge_07 + File.separator + list1[i]) &&
File.isDirectory(Nuc_merge_08 + File.separator + list2[i]))

				processFolder59(Nuc_merge_07 + File.separator + list1[i],
Nuc_merge_07 + File.separator + list1[i]);

			if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
```

```
                                processFile59(Nuc_merge_07, Nuc_merge_08,
Nuc_merge_10, list1[i], list2[i]);
                        }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile59(Nuc_merge_07, Nuc_merge_08, Nuc_merge_10, file1, file2) {
  open(Nuc_merge_07 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_merge_08 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
        saveAs("Tiff", Nuc_merge_10 + File.separator + file1);
        print("Processing: " + Nuc_merge_07 + File.separator + file1);
        print("Processing: " + Nuc_merge_08 + File.separator + file2);
        print("Saving to: " + Nuc_merge_10);
}
```

```
close("*");

//-----------------------------------------------------------------------CellProfiler Neuronal Nuclei /
```

Macro 56

```
setBatchMode(true);

processFolder42(Nuc_merge_09, Nuc_merge_10);



// function to scan folders/subfolders/files to find files with correct suffix

function processFolder42(Nuc_merge_09, Nuc_merge_10) {

	list1 = getFileList(Nuc_merge_09);

	list1 = Array.sort(list1);

	list2 = getFileList(Nuc_merge_10);

	list2 = Array.sort(list2);

	if (list1.length == list2.length) {

		for (i = 0; i < list1.length; i++) {

			if(File.isDirectory(Nuc_merge_09 + File.separator + list1[i]) &&
File.isDirectory(Nuc_merge_10 + File.separator + list2[i]))

				processFolder42(Nuc_merge_09 + File.separator + list1[i],
Nuc_merge_09 + File.separator + list1[i]);
```

```
                        if((endsWith(list1[i], suffix)) && (endsWith(list2[i], suffix)))
                                processFile42(Nuc_merge_09, Nuc_merge_10,
Nuclei_rename_1, list1[i], list2[i]);
                    }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile42(folder42, Nuc_merge_10, Nuclei_rename_1, file1, file2) {
  open(Nuc_merge_09 + File.separator + file1);
  id1 = getImageID();
  open(Nuc_merge_10 + File.separator + file2);
  id2 = getImageID();
  imageCalculator("AND ", id2, id1);
  run("Maximum...", "radius=1");
  run("Invert");

        saveAs("Tiff", Nuclei_rename_1 + File.separator + file1);
```

```
        print("Processing: " + Nuc_merge_09 + File.separator + file1);
        print("Processing: " + Nuc_merge_10 + File.separator + file2);
        print("Saving to: " + Nuclei_rename_1);
}
close("*");
//--------------------------------------------------------------------CellProfiler Neuronal Soma /
```

Macro 57

```
setBatchMode(true);
processFolder43(Soma_merge_04);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder43 (Soma_merge_04) {
  list = getFileList(Soma_merge_04);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Soma_merge_04 + File.separator + list[i]))
                processFolder43(Soma_merge_04 + File.separator + list[i]);
        if(endsWith(list[i], suffix))
                processFile43(Soma_merge_04, Soma_rename_1, list[i]);
```

```
  }
}
function processFile43 (Soma_merge_04, Soma_rename_1, file) {
  open(Soma_merge_04 + File.separator + file);
run("Adjustable Watershed", "tolerance=1.5");
run("Invert");
run("Maximum...", "radius=1");
run("Invert");


  saveAs("Tiff", Soma_rename_1 + File.separator + file);
  print("Processing: " + Soma_merge_04 + File.separator + file);
  print("Saving to: " + Soma_rename_1);
}
close("*");
//--------------------------------------------CellProfiler Neuronal Nuclei (rename function) /
```

Macro 58

```
setBatchMode(true);
processFolder80(Nuclei_rename_1);
```

```
function processFolder80 (Nuclei_rename_1) {
fList = getFileList(Nuclei_rename_1);
fList = Array.sort(fList);
for (i=0; i<fList.length; i++ ) {
                if(File.isDirectory(Nuclei_rename_1 + File.separator + fList[i]))
                processFolder80(Nuclei_rename_1 + File.separator + fList[i]);
        if(endsWith(fList[i], suffix))
                processFile80(Nuclei_rename_1, CellP_Neuronal_Nuclei, fList[i]);
}
}

function processFile80 (Nuclei_rename_1, CellP_Neuronal_Nuclei, file) {
  open(Nuclei_rename_1 + File.separator + file);
  fName = replace(fList[i], "DAPI", "Nuclei");
File.copy (Nuclei_rename_1 + File.separator + fList[i], CellP_Neuronal_Nuclei +
File.separator + fName) ;
```

```
  }

close("*");

//-----------------------------------------CellProfiler Neuronal Soma (rename function) /
```

Macro 59

```
setBatchMode(true);

processFolder81(Soma_rename_1);

function processFolder81 (Soma_rename_1) {

fList = getFileList(Soma_rename_1);

fList = Array.sort(fList);

for (i=0; i<fList.length; i++ ) {

                if(File.isDirectory(Soma_rename_1 + File.separator + fList[i]))

                processFolder81(Soma_rename_1 + File.separator + fList[i]);

        if(endsWith(fList[i], suffix))

                processFile81(Soma_rename_1, CellP_Neuronal_Soma, fList[i]);

}

}

function processFile81 (Soma_rename_1, CellP_Neuronal_Soma, file) {
```

```
  open(Soma_rename_1 + File.separator + file);

  fName = replace(fList[i], "NeuN", "Soma");

File.copy (Soma_rename_1 + File.separator + fList[i], CellP_Neuronal_Soma +

File.separator + fName) ;

  }

selectWindow("Log");

run("Close");

//------------------------------------------------------------------------------------------------------

CLOSE

//-------------------------------------------------------------------------------EMGI Score: NeuN

/ Macro 60

setBatchMode(true);

processFolder82(NeuN_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder82 (NeuN_main) {

  list = getFileList(NeuN_main);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {
```

```
        if(File.isDirectory(NeuN_main + File.separator + list[i]))
                processFolder82(NeuN_main + File.separator + list[i]);
        if(endsWith(list[i], suffix))
                processFile82(NeuN_main, list[i]);
  }
}
function processFile82 (NeuN_main, file) {
  open(NeuN_main + File.separator + file);
setAutoThreshold("Default dark");
//run("Threshold...");
setThreshold(0, 30);
                                                        // <----    Threshold value to
eliminate background in MGI Evaluation
run("Set Measurements...", "mean display redirect=None decimal=3");
run("Measure");
}
close("*");
```

```
//-------------------------------------------------------------------------------EMGI Score: DAPI
```
/ **Macro 61**

```
setBatchMode(true);

processFolder83(DAPI_main);

// function to scan folders/subfolders/files to find files with correct suffix

function processFolder83 (DAPI_main) {

  list = getFileList(DAPI_main);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(DAPI_main + File.separator + list[i]))

              processFolder83(DAPI_main + File.separator + list[i]);

        if(endsWith(list[i], suffix))

              processFile83(DAPI_main, EMGI_Results, list[i]);

  }

}

function processFile83 (DAPI_main, EMGI_Results, file) {

  open(DAPI_main + File.separator + file);

setAutoThreshold("Default dark");
```

```
//run("Threshold...");

setThreshold(0, 30);
                                                    // <----    Threshold value to
eliminate background in MGI Evaluation

run("Set Measurements...", "mean display redirect=None decimal=3");

run("Measure");

}

  saveAs("Measurements", EMGI_Results + File.separator  + "Results.csv");

selectWindow("Results");

run("Close");

//-------------------------------------------------------------------------------------------
```

CLOSE / **Macro 62**

```
beep();

Dialog.create("Finished");

Dialog.addMessage("         All Images Have Been Successfully Converted\n"

+"

                                  \n"

+"
                                                                    \n"
```

```
+"                                  ____                                    \n"
+"                                  |    |                                  \n"
+"                                  ====                                    \n"
+"                                  |    |__                                \n"
+"                                  |    |-. \\ \\                           \n"
+"                                  |__|    \\ \\                           \n"
+"                                   | |       ||                          \n"
+"                                  ====  __|                               \n"
+"                                _________ ||__                            \n"
+"                              /__  IC-DRGs  __ \\                          \n"
+"                                                                          \n"
+"
                                                                           \n"
+"
                                                                           \n"
+"   - Import/Drag these folders into CellProfiler:                         \n"
+"                                                                          \n"
+"                        *CellProfiler Neuronal Nuclei                     \n"
```

```
+"                    *CellProfiler Neuronal Soma                    \n"

+"                                                                   \n"

+"   - Import other image-sets from proteins of interest            \n"

+"                                                                   \n"

+"   - CellProfiler will use 'Neuronal Nuclei' to determine         \n"

+"     boundry for nuclei and use 'Neuronal Soma' to                \n"

+"     determine boundry of the cell body, for each neuron.         ");

Dialog.show();”
```

Appendix D – CellProfiler Pipeline

CellProfiler pipeline for using IC-DRG masks to measure nuclear, cytoplasmic, and somatic domains for 2 proteins of interest

File: IC-DRGs Template.cpproj

Github link: https://github.com/MichaelBAnderson/IC-DRGs_v1.41_FIJI_script

---

Images module: *Drag and drop image files or folders of image files into this module*

Metadata module:

Extract metadata: Yes

Metadata extraction method: Extract from file/folder names

Metadata source: File name

Regular expression to extract from file name: ^(?P<Plate>.*)_(?P<Well>[A-P][0-9](Klionsky et al.))_s(?P<Site>[0-9])_w(?P<ChannelNumber>[0-9])

Extract metadata from: All images

Metadata data type: Text

Names and Types module:

Assign a name to: Images matching rules

Process as 3D: No

Match: All (of the following rules)

Select the rule criteria: File Does Contain: **Nuclei**

Name to assign these images: **Nuclei**

Select the image type: grayscale image

Set intensity range from: Image metadata

Match: All (of the following rules)

Select the rule criteria: File Does Contain: **NeuN**

Name to assign these images: **NeuN**

Select the image type: grayscale image

Set intensity range from: Image metadata

Match: All (of the following rules)

Select the rule criteria: File Does Contain: **Protein1**

Name to assign these images: **Protein1**

Select the image type: grayscale image

Set intensity range from: Image metadata

Match: All (of the following rules)

Select the rule criteria: File Does Contain: **Protein2**

Name to assign these images: **Protein2**

Select the image type: grayscale image

Set intensity range from: Image metadata

Groups module:

Do you want to group your images: No

Pipeline module 1: Identify Primary Objects

Use advanced settings: Yes

Select the input image: **Nuclei**

Name the primary objects to be identified: **Nuclei**

Typical diameter of objects, in pixel units (Min, Max): 1 – 2000

Discard objects outside the diameter range: Yes

Discard objects touching the border of the image: Yes

Threshold strategy: Adaptive

Thresholding method: Otsu

Two-class or three-class thresholding: Three classes

Assign pixels in the middle intensity class to the foreground or the background: Foreground

Threshold smoothing scale: 1.3488

Threshold correction factor: 1.1

Lower and upper bounds on threshold: 0.0 – 1.0

Size of adaptive window: 10

Log transform before thresholding: Yes

Method to distinguish clumped objects: None

Fill holes in identified objects: After both thresholding and declumping

Handing of objects if excessive number of objects identified: Continue

Pipeline module 2: Identify Secondary Objects

Select the input image: **Soma**

Select the input objects: **Nuclei**

Name the objects to be identified: **Cells**

Select the method to identify the secondary objects: Propagation

Threshold strategy: Adaptive

Thresholding method: Otsu

Two-class or three-class thresholding: Three classes

Assign pixels in the middle intensity class to the foreground or the background: Foreground

Threshold smoothing scale: 0.0

Threshold correction factor: 1.5

Lower and upper bounds on threshold: 0.1 – 1.0

Size of adaptive window: 10

Log transform before thresholding: Yes

Regularization factor: 0.0001

Fill holes in identified objects: Yes

Discard secondary objects touching the border of the image: Yes

Discard the associated primary objects: No

Pipeline module 3: Identify Tertiary Objects

Select the larger identified objects: **Cells**

Select the smaller identified objects: **Nuclei**

Name the tertiary objects to be identified: **Cytoplasm**

Shrink smaller object prior to subtraction: Yes

Pipeline module 4: Overlay Outlines

Display outlines on a blank image: No

Select image on which to display outlines: **Protein2**

Name the output image: **OverlayImage**

Outline display mode: Color

How to outline: Thick

Select objects to display: **Cells**

Select outline color: “red”

Select objects to display: **Nuclei**

Select outline color: “green”

Pipeline module 5: Overlay Outlines

Display outlines on a black page: Yes

Name the output image: **OverlayBlank**

Outline display mode: color

How to outline: Thick

Select objects to display: **Cells**

Select outline color: “red”

Select objects to display: **Nuclei**

Select outline color: “green”

Pipeline module 6: Save Images

Select the type of image to save: Image

Select the image to save: **OverlayBlank**

Select method of constructing file names: From image filename

Select image name for file prefix: **Protein2**

Append a suffix to the image file name: No

Saved file format: jpeg

Output file location: Elsewhere (select path)

Overwrite existing files without warning: Yes

When to save: Every cycle

Record the file and path information to the saved image: No

Create subfolders in the output folder: No

Pipeline module 7: Save Images

Select the type of image to save: Image

Select the image to save: **OverlayImage**

Select method of constructing file names: From image filename

Select image name for file prefix: **Protein2**

Append a suffix to the image file name: No

Saved file format: jpeg

Output file location: Elsewhere (select path)

Overwrite existing files without warning: Yes

When to save: Every cycle

Record the file and path information to the saved image: No

Create subfolders in the output folder: No

Pipeline module 8: Measure Object Intensity

Select images to measure:

1. NeuN
2. Protein 1
3. Protein 2

Select objects to measure:

1. Cells

2. Cytoplasm
3. Nuclei

Pipeline module 9: Measure Object Size and Shape

Select object sets to measure:

1. Cells
2. Cytoplasm
3. Nuclei

Calculate the Zernike features: Yes

Calculate the advanced features: No

Pipeline module 10: Export to Spreadsheet

Select the column delimiter: Comma (“.”)

Output file location: Elsewhere (select path)

Add a prefix to the file name: Yes

Filename prefix: MyExpt_

Overwrite existing files without warning: No

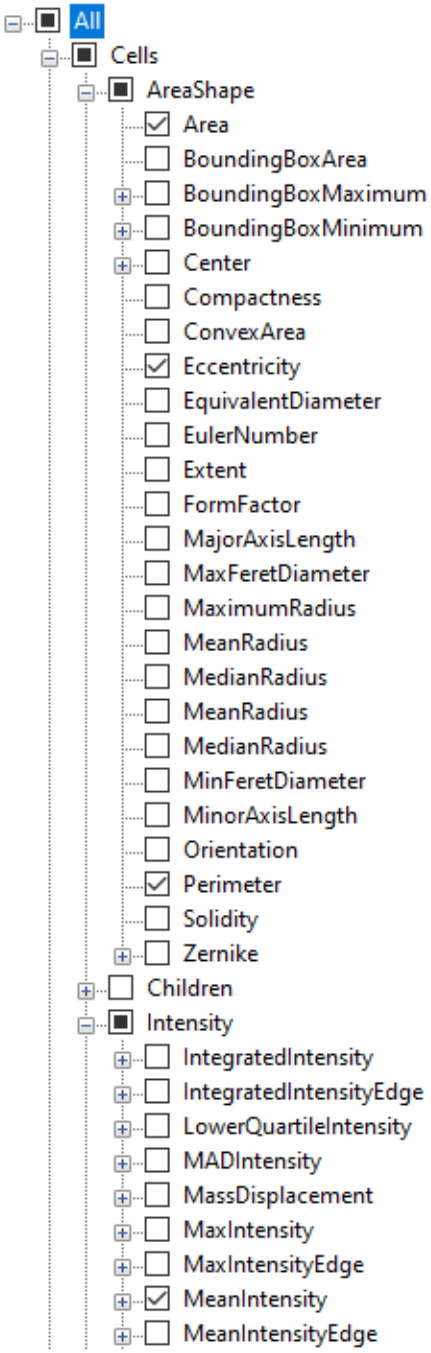
Add image metadata to columns to your object data file: No

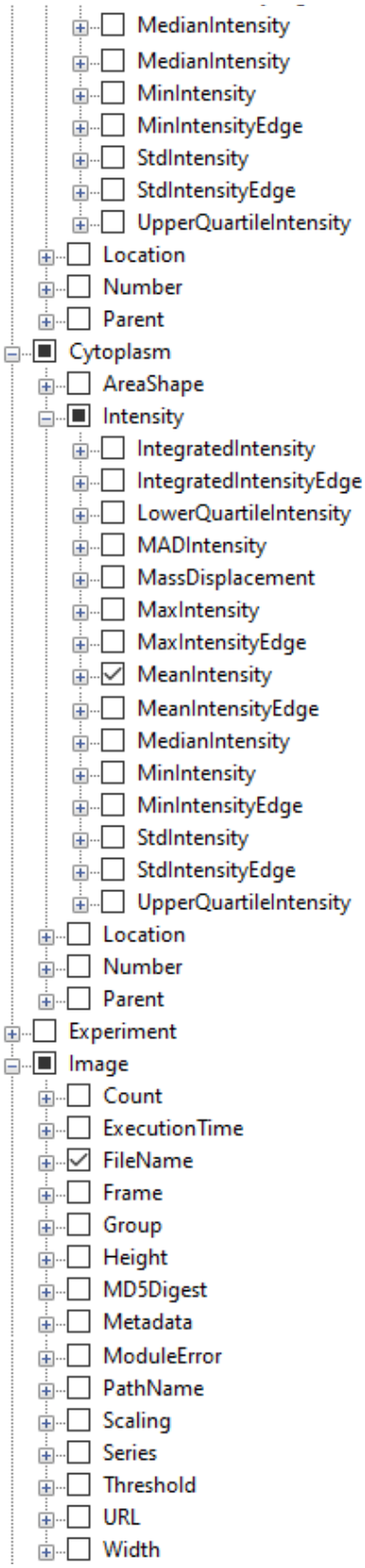
Add image file and folder names to your object data file: No

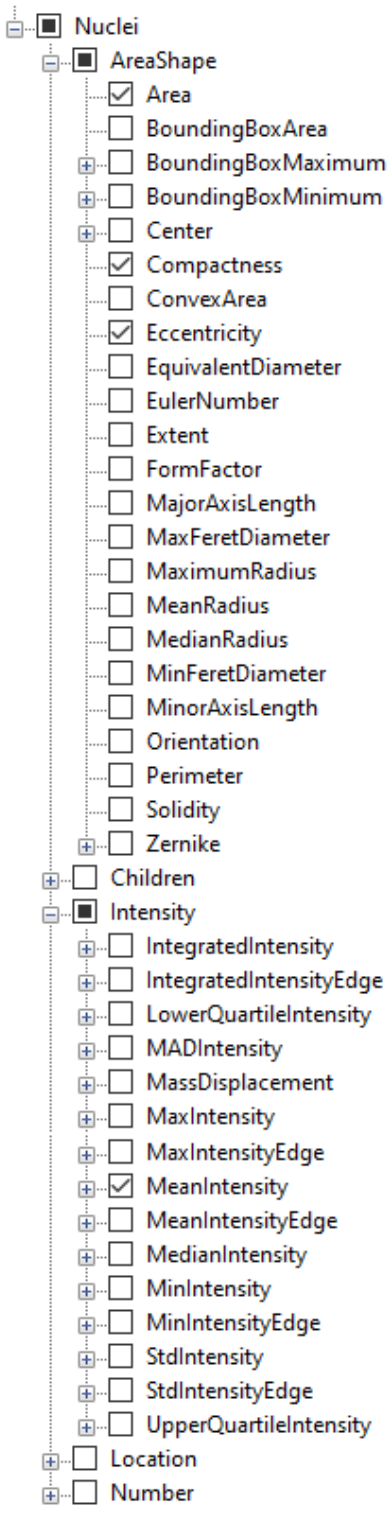
Representation of Nan/Inf: NaN

Select measurements to export: Yes

Press button to select measurements:




MedianIntensity
MedianIntensity
MinIntensity
MinIntensityEdge
StdIntensity
StdIntensityEdge
UpperQuartileIntensity
Location
Number
Parent
Cytoplasm
AreaShape
Intensity
IntegratedIntensity
IntegratedIntensityEdge
LowerQuartileIntensity
MADIntensity
MassDisplacement
MaxIntensity
MaxIntensityEdge
MeanIntensity
MeanIntensityEdge
MedianIntensity
MinIntensity
MinIntensityEdge
StdIntensity
StdIntensityEdge
UpperQuartileIntensity
Location
Number
Parent
Experiment
Image
Count
ExecutionTime
FileName
Frame
Group
Height
MD5Digest
Metadata
ModuleError
PathName
Scaling
Series
Threshold
URL
Width



Calculate the per-image mean values for object measurements: No

Calculate the per-image median values for object measurements: No

Calculate the per-image standard deviation values for object measurements: No

Create a GenePatter GCT file: No

Export all measurement types: Yes

Pipeline module 11: Export to Database

Database type: SQLite

Experiment name: MyExpt

Name of SQLite database file: DefaultDB.db

Overwrite without warning: Never

Add a prefix to table names: Yes

Table prefix: MyExpt_

Create a CellProfiler Analyst properties file: Yes

Which objects should be used for locations: **Cytoplasm**

Access CellProfiler Analyst images via URL: No

Select the plate type: None

Select the plate metadata: None

Select the well metadata: None

Include information for all images, using default values: Yes

Do you want to add group fields: No

Do you want to add filter fields: No

Select the classification type: Object

Enter a phenotype class table name if using CellProfiler too in CellProfiler Analyst: Classifer

Create a CellProfiler Analyst workspace file: No

Output file location: Elsewhere (select path)

Calculate the per-image mean values of object measurements: Yes

Calculate the per-image median values for object measurements: No

Calculate the per-image standard deviation values for object measurements: No

Export measurements for all objects to the database: Select

1. Cells
2. Cytoplasm
3. Nuclei

Export object relationships: Yes

Create one table per object, a single object table or a single object view: Single object table

Maximum # of characters in a column name: 64

Write image thumbnails directly to the database: No

Appendix E – IENF Evaluation Script

This script is operational under ImageJ/FIJI (version v1.53f +)

script begins here:

```
"//                    IENF Script Parameters

#@ File (label = "select folder: IENF Stack Repository", style = "directory") PGP

//-------------------------------------------------------------------------------------------

Operation Reminder

showMessage ("Reminder: Update the file path on Line 49, to save data.");

//------------------------------------------------------------------------------------------- Starting

Time

// showMessage ("This script measures each stack (8_Data/Statistics.xlsx) to optimize for

script accuracy.");

macro "Get Time" {

  MonthNames =

newArray("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");

  DayNames = newArray("Sun", "Mon","Tue","Wed","Thu","Fri","Sat");

  getDateAndTime(year, month, dayOfWeek, dayOfMonth, hour, minute, second, msec);
```

```
  TimeString ="Date: "+DayNames[dayOfWeek]+" ";
  if (dayOfMonth<10);}
  TimeString = TimeString+dayOfMonth+"-"+MonthNames[month]+"-"+year+"\nTime:
";
  if (hour<10);}
  TimeString = TimeString+hour+":";
  if (minute<10);}
  TimeString = TimeString+minute+":";
  if (second<10);}
  TimeString = TimeString+second;
  print(TimeString);
//-------------------------------------------------------------------------------------------------
Image Processing
setBatchMode(true);
processFolder_1(PGP);
// function to scan folders/subfolders/files to find files with correct suffix
function processFolder_1 (PGP) {
  list = getFileList(PGP);
```

```
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(PGP + File.separator + list[i]))
              processFolder_1(PGP + File.separator + list[i]);
              processFile_1(PGP, list[i]);
  }
}
function processFile_1 (PGP, file) {
  open(PGP + File.separator + file);
//  macro "Show Statistics" {
    if (nSlices>1) run("Clear Results");
    getVoxelSize(w, h, d, unit);
    n = getSliceNumber();
    for (i=1; i<=nSlices; i++) {
      setSlice(i);
      getStatistics(area, mean, min, max, std);
      row = nResults;
```

```
        if (nSlices==1)
            setResult("Area ("+unit+"^2)", row, area);
        setResult("Mean ", row, mean);
    }
    setSlice(n);
    updateResults();
IJ.renameResults("Results");
selectWindow("Results");
run("Read and Write Excel", "file=[G:/PhD Project/Manuscripts/5 - 3D IENF Manuscript (end of August)/DATA/Scripting/Mean of stack slices.xlsx]" + i);
i=1;
   if (isOpen("Results")) {
       run("Close" );
   }
   close("*");
  //saveAs("Tiff", Skeletons + File.separator + file);
 // print("Processing: " + PGP + File.separator + file);
  //print("Saving to: " + Skeletons);
```

```
}

//---------------------------------------------------------------------------------------- Closing
Time

macro "Get Time" {

  MonthNames =
newArray("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");

  DayNames = newArray("Sun", "Mon","Tue","Wed","Thu","Fri","Sat");

  getDateAndTime(year, month, dayOfWeek, dayOfMonth, hour, minute, second, msec);

  TimeString ="Date: "+DayNames[dayOfWeek]+" ";

  if (dayOfMonth<10);}

  TimeString = TimeString+dayOfMonth+"-"+MonthNames[month]+"-"+year+"\nTime:
";

  if (hour<10);}

  TimeString = TimeString+hour+":";

  if (minute<10);}

  TimeString = TimeString+minute+":";

  if (second<10);}

  TimeString = TimeString+second;
```

```
  print(TimeString);

}

  //-------------------------------------------------------------------------------------------------
- Log Save

selectWindow("Log");

save(Data + "/" +"Log.txt");

run("Close");

showMessage ("IENF Processing Complete");”
```

Appendix F – IENF Dimming Script

This script is operational under ImageJ/FIJI (version v1.53f +)

script begins here:

```
“//                    IENF Script Parameters

#@ File (label = "select folder: to Dim", style = "directory") to_Dim

#@ File (label = "select folder: Dimmed", style = "directory") Dimmed

//------------------------------------------------------------------------------------------ Starting
Time

// showMessage ("This script measures each stack (8_Data/Statistics.xlsx) to optimize for
script accuracy.");

macro "Get Time" {

  MonthNames =
newArray("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");

  DayNames = newArray("Sun", "Mon","Tue","Wed","Thu","Fri","Sat");

  getDateAndTime(year, month, dayOfWeek, dayOfMonth, hour, minute, second, msec);

  TimeString ="Date: "+DayNames[dayOfWeek]+" ";
```

```
  if (dayOfMonth<10);}
  TimeString = TimeString+dayOfMonth+"-"+MonthNames[month]+"-"+year+"\nTime:
";
  if (hour<10);}
  TimeString = TimeString+hour+":";
  if (minute<10);}
  TimeString = TimeString+minute+":";
  if (second<10);}
  TimeString = TimeString+second;
  print(TimeString);
//---------------------------------------------------------------------------------------- Dimming
macro
setBatchMode(true);
processFolder_1(to_Dim);
function processFolder_1 (to_Dim) {
  list = getFileList(to_Dim);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
```

```
        if(File.isDirectory(to_Dim + File.separator + list[i]))
                processFolder_1(to_Dim + File.separator + list[i]);
                processFile_1(to_Dim, Dimmed, list[i]);
  }
}
function processFile_1 (to_Dim, Dimmed, file) {
  open(to_Dim + File.separator + file);
//run("Brightness/Contrast...");
setMinAndMax(20, 275);
run("Apply LUT", "stack");
  saveAs("Tiff", Dimmed + File.separator + file);
  print("Processing: " + to_Dim + File.separator + file);
  print("Saving to: " + Dimmed);
close("*");
}
close("*");
//------------------------------------------------------------------------------------ Closing
Time
```

```
macro "Get Time" {
   MonthNames =
newArray("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");
   DayNames = newArray("Sun", "Mon","Tue","Wed","Thu","Fri","Sat");
   getDateAndTime(year, month, dayOfWeek, dayOfMonth, hour, minute, second, msec);
   TimeString ="Date: "+DayNames[dayOfWeek]+" ";
   if (dayOfMonth<10);}
   TimeString = TimeString+dayOfMonth+"-"+MonthNames[month]+"-"+year+"\nTime:
";
   if (hour<10);}
   TimeString = TimeString+hour+":";
   if (minute<10);}
   TimeString = TimeString+minute+":";
   if (second<10);}
   TimeString = TimeString+second;
   print(TimeString);
}
```

```
//-----------------------------------------------------------------------------------------------------
- Log Save

selectWindow("Log");

save(Data + "/" +"Log.txt");

run("Close");

showMessage ("IENF Processing Complete");”
```

Appendix G – IENF Resolver Script

Intra-Epidermal Nerve Fiber (IENF) 63x Consolidation Script version 0.14

File: 63x_IENF_Script_revision_063_(Home_version).ijm

Github link:

This script is operational under ImageJ/FIJI (version v1.53f +)

script begins here:

"// IENF script for three-dimensional imaging

#@ File (label = "select folder 1: PGP 9.5", style = "directory") PGP

#@ File (label = "select folder 2: Alpha_blanks", style = "directory") Alpha_Blanks

#@ File (label = "select folder 3: Omega_blanks", style = "directory") Omega_Blanks

#@ File (label = "select folder 4: Skeletons", style = "directory") Skeletons

#@ File (label = "select folder 5: Mask_overlay_1", style = "directory") Mask_overlay_1

#@ File (label = "select folder 6: Mask overlay 2", style = "directory") Mask_Overlay_2

#@ File (label = "select folder 7: Binary folder", style = "directory") Binary

#@ File (label = "select folder 8: Numbered mesh", style = "directory") Numbered_Mesh

#@ File (label = "select folder 9: Numbered masks for measurement", style = "directory") Numbered_Masks_for_measurement

#@ File (label = "select folder 10: IENF scoring", style = "directory") IENF_Scoring

#@ File (label = "select folder 11: Final processing", style = "directory") Final_Processing

#@ File (label = "select folder 12: Data", style = "directory") Data

// Revisions

// Previous: ALL previous versions of the 3D-IENF script were designed for a 40x objective. Excessive consolidation of IENF structures appear to occur at the higher than normal a threshold of 75-255.. try a threshold of 100-255

// Revision 001: changed threshold in "Image Processing" macro from "75-255" to "100-255" (result: better but missing a few nerves. i think it is due to 3D Counter size limitation. will change min from 1000 to 500)

// Revision 002: changed 3D object counter size limit from 1000 to 500 (result: better but still missing this nerves, lower 3D Counter size to lower limit to 200)

// Revision 003: changed 3D object counter size limit from 500 to 200 (result: not much difference but will keep it for now). lower threshold

// Revision 004: lowered threshold from 100 to 90 (result: looks much better, going to lower to test)

// Revision 005: lowered threshold from 90 to 83 (result: better but missing dim fibers, ex. 63x_IS-02)

// Revision 006: multiple variables: adding de-speckle and lowering threshold from 83 to 60. i also added invert/particle analyzer on skeletons (to limit clustering) in "Image Processing" macro (result: looks very good, need to increase 3D Object Counter minimum size limit from 200 to the 1000)

// Revision 007: increased 3D Object Counter minimum size limit from 200 to 1000 (result: missing a few thin fibers, lowering 3D Object Counter minimum size limit from 1000 to 500 in rev008)

// Revision 008: reduced 3D Object Counter minimum size limit from 1000 to 500. (result: I am at work and only running one IS at a time but on IS-02, this looks great, all whole and 2 fragments, the fragments are two portions of one nerve and lacked IR in the middle of the Z section).

// Note: So far, this is looking outstanding. I am at work but this is worth the time running IS-1-5

// Revision 009: updated file path for saving .csv files, running with IS-1-5 (result: looks amazing for image stacks with little stratum basal non-specific IR) *consider raising threshold

// Revision 010: increased threshold from 60 to 70, also updated 1st analyze skeleton size limits from 0-25 (from 0-20) to make compatible with second analyze

particles (result: Revision 010 is slightly better but too dim, also stratum basal IR is leading to clumping)

// NOTE: Some image sets have excessive stratum basal IR (ex. IS_01 and IS_05) and when these are identified, use: "Image -> Stack -> Delete Slice Range" to delete slices excessive IR

// NOTE: So far, I deleted slices 0-25 (IS-01) and slices 0-68 (IS-05)

// Revision 011: Decreased threshold from 70 to 65, running resliced IS-01 and IS-05 (result: looks great but some clumping)

// Revision 012: Increasing threshold from 65 to 75, Testing to see if a higher threshold works better for reducing clumping (result: this is by far the best script... nearly all are accurate. now, see how this works for 40x. saving backup version of this. TOO much non-specific background in some image stacks (ex. 63x__IS-06 and 07). Processing IS_06 crashed the software.

// Revision 013: increasing threshold from 75 to 85. Increasing 3D Object Counter size threshold from 500 to 2000

// Revision 014: Adding resize function to change 1024 image stacks to 512x512 for speed and consistancy (result: the FOV was resized before setting scale and resulted in 1/2 size scale

// NOTE: Okay.. many 40x images are actualy 64x and vice versa, fixing now.

// Revision 015: removing the resize and scale setting functions to test compatibility

// Revision 016: Lowering 3D object counter minimum from 2000 to 500 (missed nerves)

// NOTE: This looks great on 512x512 image-sets with different Zoom factors. There are a few skinny fibers that are still lost due to 3D Object Counter. Reducing 3D OC min from 500 to 250

// Revision 017: Reducing 3D OC min from 500 to 250

// Revision 018: Increased second instance of Mask of Nearby Points, from 2 to 2.5

// Revision 019: Looking good but missing dim structures. Decreasing threshold from 85 to 65

// Revision 020: Havent checked 0.19 yet but am running a min Threshold of 75 for comparision

// Revision 021: Reduced second instance of Mask of Nearby Points, from 2.5 to 2.0

// Revision 022: Increasing both instances of Mask of Nearby Points, from 2.0 to 3.0

// Revision 023: decreasing minimum size from 250 to 100

// Revision 024: decreasing threshold from 75 to 65, increasing 3D OC from 100 to 500

// Revision 025: decreasing threshold from 65 to 50, reducing mask of nearby point value from 3 to 1.5

// Revision 026: removing 2nd "despeckle", adding three sequential instances of "max (3D)", increased 3D OC minimum size from 500 to 2000

// NOTE: This looks much better but there is slight clumping.

// Revision 027: removed the "Maximum (3D)" function and added "Max (3D)". This is in hopes of reducing IENF clumping while keeping quality.

// NOTE: looks very good but slight clumping

// Revision 028: removed one instance of "Max (3D)" to reduce clumping

// NOTE: looks very good but still slight clumping

// Revision 029: removed one more instance of "Max (3D)" to reduce clumping (carefully evaluate results, 028 may be best)

// NOTE: Looks great in dim images but there is excessive clumping in bright images. Try to find a good balance with threshold (60). losing small nerves. decrease 3D OC minimum from 2000 to 1000

// Revision 030: increasing threshold from 50 to 60, lowering 3D OC min size from 2000 to 1000, removed analyze particles (unneeded), increased second mask of nearby points to 3. removed "invert"

// Revision 031: looks good for bright FOV but too dim in dim images. lowering threshold from 60 to 50

// NOTE: Really looks good. I will try to lower threshold to get more dim IENFs

// Revision 032: reduced threshold from 50 to 45

// NOTE: Great for these two images. Try more FOVs

// Revision 033: this is a backup of Revision 0.32 (duplicate of 0.32)

// NOTE: This looks great on some FOVs but there is large areas of clunping present in some FOVs with mild non-specific background

// Revision 034: added 3D Hysteresis Thresholding to remove background

// Revision 035: changing 3D Hysteresis Thresholding from 128-50 to 128-25

// Revision 036: experimenting: lowering mask of nearby points, added max (3D)

// NOTE: 3D hysteresis removes background well but I also loses dim, superficial fibers. I tried for a few hours to use 3D hysteresis but I continued to miss the superficial tips of IENFs

// STATEMENT: Bright images have brighter background fluorescence than dim images. Dim fibers from dim images are the same level of intensity as background (mean:13) in bright images. No way to indescriminate on threshold.

// Develop a method of getting mean from every image stack slice so user can delete stack slices above a certain value (probably 12) with the slice remover. Standardization.

// NOTE: I created a separate script for gathering mean from stack slices.

// STATEMENT: This worked great and here are the rules:

// RULE 1: Highlight in Excel sheet the and delete stack slices that have a mean of 12 or greater.

// RULE 2: Calculate the mean of the stack in Excel and process any stack with an overall MGI above 10 by the to be made (TBM) script to lower Contrast/Brightness

// Revision 037: Tried to implement the stack slice mean script (for stack evaluation) into this script but it was clunky and best suited independent.

// Revision 038: Duplicate of the best version so far (0.32)

// Revision 039: Adding a weak Brightness/Contrast element for subtle sharpness (final changes, hopefully)

// Revision 040: FINAL VERSION: duplicate of version 0.39. Added scales used in study (below, 1-4)

// NOTE: I removed the scale to make it universal, but the "Mask of Nearby Points" function losses scale data

// SETTING SCALE: Scale must be set after "Masks of Nearby Points". Branching into scale specific

// Revision 041: Adding "Set Scale" function after macro 1. The scale will need to be updated for different resolutions

// Image resolution is either encoded in the image itself or can be calculated from microscope camera.

// Below are examples of the 4 scales used in this study:

// Revision 042: added contrast (110, 275) but resulted in error. may have worked with higher threshold than 45

// Revision 043: Duplicate of the best version, so far, revision 41.

// Revision 044: Inserted "add slices" (3X) to beginning of stack and "add slices" (3X) at end of stack

// NOTE: need to add new folder, so that images have the same number of slices for the image calculator

// Revision 045: Added "Alpha_Blanks" and "Omega_Blanks" folder/macro to insert 5 blanks at the beginnging and 5 blanks at the end of the stack before main processing (macro 3) and combining with image calculator (macro 4). Excluding on Edges in 3D OC Counter

// Revision 046: Fixed the insertation of blanks at beginning and end of stack

// Revision 047: Removing an unneed (Main_Processing) folder

// Revision 048: Inserting only one image into start/end of stack, instead of 5 (seeing if it works, this will prevent skewing of establish mean threshold rules)

// NOTE: WORKED!! This is the final version

// Revision 049: duplicate of version 048.

// Revision 050: added extra Mask of Nearby Points and it seems to look better, compare with 0.49 data

// NOTE: too dim, fragmentation

// Revision 051: lowered threshold from 75 back to 45, added analyze particles (size 2 worked well, testing with 512)

// Revision 052: removed "Mask of nearby points" because it removes scale and replaced it with something better that also retains scale

// NOTE: Dilate3D dilated too much in Z

// Revision 053: added 2D dilation and only 2 instances of dilate3D

// Revision 054: removing the two instances of dilation3D to test results

// NOTE: Revision 0.52 is the best. The binary and 3D dilation seems to stretch the geometry

// Revision 055: duplicate of version 052

// Revision: 056: decreased 3D object counter from 1000 to 500

// NOTE: better with limit to 1000

// Revision 057: increased 3D object counter from 500 to 1000

// Revision 058: duplicate of revision 057

// NOTE: "Mask of Nearby Points" plugin is unique. It loses the scale and replaces IR with assigned circular points that skrinks the objects down and makes it easier to score

// Revision 059 (512x512 FINAL): added "Mask of Nearby Points" function for "Numbered Masks"

// NOTE: with new knowledge about "mask of nearby points" I created new folders for viewing mesh.

// Mask_overlay_1: Image calculator (ADD) of Binary and Skeletons stacks and processed by "Mask of Nearby Points" (for scoring)

// Mask_Overlay_2: Image calculator (ADD) of Binary and Skeletons stacks (for measurement)

// Numbered_Mesh: Binary folder processsed by 3D Object Counter, numbered/segmented objects, and processed by "Mask of Nearby Points" (for scoring)

// Numbered_Masks_for_measurement: Binary folder processsed by 3D Object Counter and numbered/segmented objects (for measurement)

//Revision 064: renamed folders, FINAL version

//---

Operation Reminders

// showMessage ("Use IENF Stack Evaluation script to determine if mean of each stack slice less than 12. Calculate mean of stack to determine it is less than 11, if so, run through Dimmer script, to reduce intensity");

// showMessage ("If these steps are taken then accuracy of positive identification should be close to 95%");

```
//------------------------------------------------------------------------------------------ Starting
Time
macro "Get Time" {
  MonthNames =
newArray("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");
  DayNames = newArray("Sun", "Mon","Tue","Wed","Thu","Fri","Sat");
  getDateAndTime(year, month, dayOfWeek, dayOfMonth, hour, minute, second, msec);
  TimeString ="Date: "+DayNames[dayOfWeek]+" ";
  if (dayOfMonth<10);}
  TimeString = TimeString+dayOfMonth+"-"+MonthNames[month]+"-"+year+"\nTime:
";
  if (hour<10);}
  TimeString = TimeString+hour+":";
  if (minute<10);}
  TimeString = TimeString+minute+":";
  if (second<10);}
  TimeString = TimeString+second;
  print(TimeString);
```

```
//------------------------------------------------------------------------------------------ - - - - - - -
-   Alpha Blanks - Adding 5 blanks to beginning of stack (macro 01)
setBatchMode(true);
processFolder_1(PGP);
function processFolder_1 (PGP) {
  list = getFileList(PGP);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(PGP + File.separator + list[i]))
                processFolder_1(PGP + File.separator + list[i]);
                processFile_1(PGP, Alpha_Blanks, list[i]);
  }
}
function processFile_1 (PGP, Alpha_Blanks, file) {
  open(PGP + File.separator + file);
setSlice(1);
run("Add Slice");
setSlice(1);
```

```
run("Select All");

run("Copy");

setSlice(2);

run("Paste");

setSlice(1);

run("Select All");

setBackgroundColor(0, 0, 0);

run("Clear", "slice");

  saveAs("Tiff", Alpha_Blanks + File.separator + file);

  print("Processing: " + PGP + File.separator + file);

  print("Saving to: " + Alpha_Blanks);

close("*");

}

//------------------------------------------------------------------------------------------ - - - - - -  -

-   Omega Blanks - Adding 5 blanks to end of stack (macro 02)

setBatchMode(true);

processFolder_2(Alpha_Blanks);

function processFolder_2 (Alpha_Blanks) {
```

```
  list = getFileList(Alpha_Blanks);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Alpha_Blanks + File.separator + list[i]))
              processFolder_2(Alpha_Blanks + File.separator + list[i]);
              processFile_2(Alpha_Blanks, Omega_Blanks, list[i]);
  }
}
function processFile_2 (Alpha_Blanks, Omega_Blanks, file) {
  open(Alpha_Blanks + File.separator + file);
Stack.setSlice (nSlices);
run ("Add Slice");
  saveAs("Tiff", Omega_Blanks + File.separator + file);
  print("Processing: " + Alpha_Blanks + File.separator + file);
  print("Saving to: " + Omega_Blanks);
close("*");
}
```

```
//------------------------------------------------------------------------------------------ - - - - - - -
-   Main Processing (macro 03)
setBatchMode(true);
processFolder_3(Omega_Blanks);
function processFolder_3 (Omega_Blanks) {
  list = getFileList(Omega_Blanks);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Omega_Blanks + File.separator + list[i]))
                processFolder_3(Omega_Blanks + File.separator + list[i]);
                processFile_3(Omega_Blanks, Skeletons, list[i]);
  }
}
function processFile_3 (Omega_Blanks, Skeletons, file) {
  open(Omega_Blanks + File.separator + file);
run("Despeckle", "stack");
setAutoThreshold("Default dark");
setThreshold(45, 255);
```

```
setOption("BlackBackground", false);

run("Convert to Mask", "method=Default background=Dark");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Skeletonize (2D/3D)");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");
```

```
run("Dilate (3D)", "iso=255");

run("Skeletonize (2D/3D)");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Dilate (3D)", "iso=255");

run("Skeletonize (2D/3D)");

  saveAs("Tiff", Skeletons + File.separator + file);

  print("Processing: " + Omega_Blanks + File.separator + file);

  print("Saving to: " + Skeletons);

close("*");

}

//------------------------------------------------------------------------------------------ - - - - - -  -

-  Mask Overlay Scoring (macro 04)

setBatchMode(true);
```

```
processFolder_4(Omega_Blanks, Skeletons);



function processFolder_4 (Omega_Blanks, Skeletons) {
  list1 = getFileList(Omega_Blanks);
  list1 = Array.sort(list1);
  list2 = getFileList(Skeletons);
  list2 = Array.sort(list2);
        if (list1.length == list2.length) {
                for (i = 0; i < list1.length; i++) {
                        if(File.isDirectory(Omega_Blanks + File.separator + list1[i]) &&
File.isDirectory(Skeletons + File.separator + list2[i]))
                                processFolder_4(Omega_Blanks + File.separator + list1[i],
Omega_Blanks + File.separator + list1[i]);
                                processFile_4(Omega_Blanks, Skeletons, Mask_overlay_1,
list1[i], list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
```

```
}

}

function processFile_4(Omega_Blanks, Skeletons, Mask_overlay_1, file1, file2) {

open(Omega_Blanks + File.separator + file1);

 id1 = getImageID();

 open(Skeletons + File.separator + file2);

 id2 = getImageID();

 imageCalculator("Add create stack", id2, id1);

 run("Despeckle", "stack");

setThreshold(25, 255);

setOption("BlackBackground", false);

run("Convert to Mask", "method=Default background=Light");

run("Mask Of Nearby Points", "add=.0001 ...=128");

        saveAs("Tiff", Mask_overlay_1 + File.separator + file1);

        print("Processing: " + Omega_Blanks + File.separator + file1);

        print("Processing: " + Skeletons + File.separator + file2);

        print("Saving to: " + Mask_overlay_1);
```

```
close("*");

}

//------------------------------------------------------------------------------------------ - - - - - - -

-   Mask Overlay Measurement (macro 04)

setBatchMode(true);

processFolder_5(Omega_Blanks, Skeletons);

function processFolder_5 (Omega_Blanks, Skeletons) {

  list1 = getFileList(Omega_Blanks);

  list1 = Array.sort(list1);

  list2 = getFileList(Skeletons);

  list2 = Array.sort(list2);

        if (list1.length == list2.length) {

                for (i = 0; i < list1.length; i++) {

                        if(File.isDirectory(Omega_Blanks + File.separator + list1[i]) &&
File.isDirectory(Skeletons + File.separator + list2[i]))

                                processFolder_5(Omega_Blanks + File.separator + list1[i],
Omega_Blanks + File.separator + list1[i]);
```

```
                    processFile_5(Omega_Blanks, Skeletons,
Mask_Overlay_2, list1[i], list2[i]);
                }
        } else {
        print("The two directories do not have the same # of files!");
}
}
function processFile_5(Omega_Blanks, Skeletons, Mask_Overlay_2, file1, file2) {
open(Omega_Blanks + File.separator + file1);
 id1 = getImageID();
 open(Skeletons + File.separator + file2);
 id2 = getImageID();
 imageCalculator("Add create stack", id2, id1);
        saveAs("Tiff", Mask_Overlay_2 + File.separator + file1);
        print("Processing: " + Omega_Blanks + File.separator + file1);
        print("Processing: " + Skeletons + File.separator + file2);
        print("Saving to: " + Mask_Overlay_2);
```

```
close("*");

}

//------------------------------------------------------------------------------------------ - - - - - - -
-   Binary for measurement (macro 05)

setBatchMode(true);

processFolder_6(Mask_Overlay_2);

function processFolder_6 (Mask_Overlay_2) {

  list = getFileList(Mask_Overlay_2);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {

        if(File.isDirectory(Mask_Overlay_2 + File.separator + list[i]))

              processFolder_6(Mask_Overlay_2 + File.separator + list[i]);

              processFile_6(Mask_Overlay_2, Binary, list[i]);

  }

}

function processFile_6(Mask_Overlay_2, Binary, file) {

  open(Mask_Overlay_2 + File.separator + file);

setAutoThreshold("Default dark");
```

```
//run("Threshold...");

setThreshold(75, 255);

setOption("BlackBackground", false);

run("Convert to Mask", "method=Default background=Dark");

run("16-bit");  // this function is important for the thresholding and isolation of individual nerves from 3D Object Counter's "Object" output image

  saveAs("Tiff", Binary + File.separator + file);

  print("Processing: " + Mask_Overlay_2 + File.separator + file);

  print("Saving to: " + Binary);

close("*");

}

//------------------------------------------------------------------------------------------ - - - - - -  -

-   3D Object Counter for Numbered Masks for Scoring (macro 6)

setBatchMode(true);

processFolder_7(Binary);

function processFolder_7 (Binary) {

  list = getFileList(Binary);

  list = Array.sort(list);
```

```
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Binary + File.separator + list[i]))
            processFolder_7(Binary + File.separator + list[i]);
        processFile_7(Binary, Numbered_Mesh, list[i]);
  }
}
function processFile_7 (Binary, Numbered_Mesh, file) {
  open(Binary + File.separator + file);
run("3D OC Options", "volume surface nb_of_obj._voxels nb_of_surf._voxels
integrated_density mean_gray_value std_dev_gray_value median_gray_value
minimum_gray_value maximum_gray_value centroid mean_distance_to_surface
std_dev_distance_to_surface median_distance_to_surface centre_of_mass bounding_box
close_original_images_while_processing_(saves_memory) dots_size=80 font_size=80
show_numbers white_numbers
store_results_within_a_table_named_after_the_image_(macro_friendly)
redirect_to=none");
run("3D Objects Counter", "threshold=80 slice=110 min.=1000 max.=63700992
exclude_objects_on_edges objects surfaces");
setAutoThreshold("Default dark");
```

```
run("8-bit");

run("Find Edges", "stack");

setMinAndMax(0, 0);

run("Apply LUT", "stack");

run("Mask Of Nearby Points", "add=.0001 ...=128");

  saveAs("Tiff", Numbered_Mesh + File.separator + file);

  print("Processing: " + Binary + File.separator + file);

  print("Saving to: " + Numbered_Mesh);

close("*");

}

//------------------------------------------------------------------------------------------ - - - - - - -

-  3D Object Counter for Numbered Masks for Mesh (macro 6)

setBatchMode(true);

processFolder_8(Binary);

function processFolder_8 (Binary) {

  list = getFileList(Binary);

  list = Array.sort(list);

  for (i = 0; i < list.length; i++) {
```

```
        if(File.isDirectory(Binary + File.separator + list[i]))
                processFolder_8(Binary + File.separator + list[i]);
                processFile_8(Binary, Numbered_Masks_for_measurement, list[i]);
  }
}
function processFile_8 (Binary, Numbered_Masks_for_measurement, file) {
  open(Binary + File.separator + file);
run("3D OC Options", "volume surface nb_of_obj._voxels nb_of_surf._voxels
integrated_density mean_gray_value std_dev_gray_value median_gray_value
minimum_gray_value maximum_gray_value centroid mean_distance_to_surface
std_dev_distance_to_surface median_distance_to_surface centre_of_mass bounding_box
close_original_images_while_processing_(saves_memory) dots_size=80 font_size=80
show_numbers white_numbers
store_results_within_a_table_named_after_the_image_(macro_friendly)
redirect_to=none");
run("3D Objects Counter", "threshold=80 slice=110 min.=1000 max.=63700992
exclude_objects_on_edges objects surfaces");
setAutoThreshold("Default dark");
run("8-bit");
```

```
run("Find Edges", "stack");

setMinAndMax(0, 0);

run("Apply LUT", "stack");

  saveAs("Tiff", Numbered_Masks_for_measurement + File.separator + file);

  print("Processing: " + Binary + File.separator + file);

  print("Saving to: " + Numbered_Masks_for_measurement);

close("*");

}

//------------------------------------------------------------------------------------------ - - - - - -  -

-  IENF Scoring (macro 7)

setBatchMode(true);

processFolder_9(Mask_overlay_1, Numbered_Mesh);

function processFolder_9 (Mask_overlay_1, Numbered_Mesh) {

list1= getFileList(Mask_overlay_1);

n1=lengthOf(list1);

print("n1 = ",n1);

list2= getFileList(Numbered_Mesh);

n2=lengthOf(list2);
```

```
small = n1;
if(small<n2)
        small = n2;
for(i=0;i<small;i++)
   {
        stack1=list1[i];
        stack2=list2[i];
   function processFile_9(Mask_overlay_1, Numbered_Mesh, IENF_Scoring) {
   }
   open(Mask_overlay_1+File.separator+list1[i]);
newName1 = "image-" + list1[i];
   rename(newName1);
   open(Numbered_Mesh+File.separator+list2[i]);
   newName2 = "overlay-" + list1[i];
   rename(newName2);
        print("processing image",i);
        run("Merge Channels...", "c1=" + newName1 + " c7=" + newName2);
```

```
        saveAs("Tiff", IENF_Scoring + File.separator + list1[i]);
        print("Processing: " + Mask_overlay_1 + File.separator + Mask_overlay_1);
        print("Processing: " + Numbered_Mesh + File.separator + list2[i]);
        print("Saving to: " + IENF_Scoring);
close("*");
}
//------------------------------------------------------------------------------------------ - - - - - -  -
-   3D Object Counter for Skeletons (macro 8)
setBatchMode(true);
processFolder_10(Binary);
function processFolder_10 (Binary) {
  list = getFileList(Binary);
  list = Array.sort(list);
  for (i = 0; i < list.length; i++) {
        if(File.isDirectory(Binary + File.separator + list[i]))
                processFolder_10(Binary + File.separator + list[i]);
                processFile_10(Binary, Final_Processing, list[i]);
  }
```

```
}

function processFile_10 (Binary, Final_Processing, file) {

  open(Binary + File.separator + file);

run("3D OC Options", "volume surface nb_of_obj._voxels nb_of_surf._voxels
integrated_density mean_gray_value std_dev_gray_value median_gray_value
minimum_gray_value maximum_gray_value centroid mean_distance_to_surface
std_dev_distance_to_surface median_distance_to_surface centre_of_mass bounding_box
close_original_images_while_processing_(saves_memory) dots_size=5 font_size=10
store_results_within_a_table_named_after_the_image_(macro_friendly)
redirect_to=none");

run("3D Objects Counter", "threshold=80 slice=121 min.=1000 max.=63700992
exclude_objects_on_edges objects statistics summary");

IJ.renameResults("Results");

selectWindow("Results");

run("Read and Write Excel", "file=[G:/PhD Project/Manuscripts/5 - 3D IENF Manuscript
(end of August)/DATA/Scripting/IENF Scripting Area/12_Data/IENF_Data.xlsx]" + i);
// update this for reminder message on last revision

i=1;

   if (isOpen("Results")) {

       run("Close" );
```

```
    }
run("Skeletonize (2D/3D)");
run("Analyze Skeleton (2D/3D)", "prune=[shortest branch] prune_0 calculate");
selectWindow("Results");
run("Read and Write Excel", "file=[G:/PhD Project/Manuscripts/5 - 3D IENF Manuscript (end of August)/DATA/Scripting/IENF Scripting Area/12_Data/IENF_Data.xlsx]"); // update this for reminder message on last revision
i=1;
    if (isOpen("Results")) {
        run("Close" );
    }
   saveAs("Tiff", Final_Processing + File.separator + file);
   print("Processing: " + Binary + File.separator + file);
   print("Saving to: " + Final_Processing);
close("*");
}
//------------------------------------------------------------------------------------------ Closing Time
```

```
macro "Get Time" {
   MonthNames =
newArray("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");
   DayNames = newArray("Sun", "Mon","Tue","Wed","Thu","Fri","Sat");
   getDateAndTime(year, month, dayOfWeek, dayOfMonth, hour, minute, second, msec);
   TimeString ="Date: "+DayNames[dayOfWeek]+" ";
   if (dayOfMonth<10);}
   TimeString = TimeString+dayOfMonth+"-"+MonthNames[month]+"-"+year+"\nTime:
";
   if (hour<10);}
   TimeString = TimeString+hour+":";
   if (minute<10);}
   TimeString = TimeString+minute+":";
   if (second<10);}
   TimeString = TimeString+second;
   print(TimeString);
   //-----------------------------------------------------------------------------------------------
- Log Save
```

```
selectWindow("Log");

save(Data + "/" +"Log.txt");

run("Close");

showMessage ("IENF Processin1.1g Complete");”
```

VITA

Michael Brian Anderson

Candidate for the Degree of

Doctor of Philosophy

Thesis: A NEW AGE: QUANTITATIVE MORPHOMETRY FOR THE 21ST CENTURY – ADVANCES IN THE STUDY OF DORSAL ROOT GANGLION

Major Field: Biomedical Sciences

Biographical:

Completed the requirements for the Doctor of Philosophy in Biomedical Sciences at Oklahoma State University Center for Health Sciences, Tulsa, Oklahoma in December, 2021.

Education: Received Bachelor of Science degree in Molecular Biology from Northeastern State University.

Experience: Raised in Tulsa, Oklahoma; employed by Oklahoma State University Center for Health Sciences as a research assistant, 2008 – to present.

Professional Memberships: Society for Neuroscience (current), Oklahoma Microscopy Society (student representative member, current), Journal of Online Video Experimentation (JOVE, current), Microscopy Society of America, American Association of Anatomists, Microanalysis Society, and Biomedical Sciences Graduate Student Association (BSGSA), Treasurer (2018-2019).