2018

APPLICATIONS OF CELL-DERIVED VESICLES: FROM SINGLE MOLECULE STUDIES TO DRUG DELIVERY

Faruk H. Moonschi
University of Kentucky, faruk.moonschi@gmail.com
Digital Object Identifier: https://doi.org/10.13023/ETD.2018.217

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Faruk H. Moonschi, Student

Dr. Christopher I. Richards, Major Professor

Dr. Mark A. Lovell, Director of Graduate Studies

APPLICATIONS OF CELL-DERIVED VESICLES: FROM SINGLE MOLECULE STUDIES TO DRUG DELIVERY

DISSERTATION

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the College of Arts and Sciences at the University of Kentucky

By
Faruk Hossain Moonschi
Lexington, Kentucky

Director: Christopher I. Richards, Assistant Professor of Chemistry
Lexington, Kentucky
2018

Abstract of Dissertation

APPLICATIONS OF CELL-DERIVED VESICLES: FROM SINGLE MOLECULE STUDIES TO DRUG DELIVERY

Single molecule studies can provide information of biological molecules which otherwise is lost in ensemble studies. A wide variety of fluorescence-based techniques are utilized for single molecule studies. While these tools have been widely applied for imaging soluble proteins, single molecule studies of transmembrane proteins are much more complicated. A primary reason for this is that, unlike membrane proteins, soluble proteins can be easily isolated from the cellular environment. One approach to isolate membrane proteins into single molecule level involves a very low label expression of the protein in cells. However, cells generate background fluorescence leading to a very low signal to noise ratio. An alternative approach involves isolating membrane proteins in artificial membrane derived vesicles. This approach is limited to proteins which can be solubilized or stabilized in detergent solution. This intermediate step endangers the structural integrity of proteins with multiple subunits. Hence, we isolated transmembrane proteins into cell-derived vesicles which maintain the proteins in their physiological membrane without compromising their functional integrity. We studied the stoichiometric assembly of $\alpha3\beta4$ nicotinic receptors which are pentameric receptor with possible stoichiometry of $(\alpha3)_2(\beta4)_3$ and $(\alpha3)_3(\beta4)_2$. We found that $(\alpha3)_2(\beta4)_3$ is the predominant stoichiometry, and we have verified our finding with both single and double color experiments. We have also demonstrated that cell-derived vesicles can be utilized to study ligand receptor interactions.

Cell-derived vesicles generated from cellular preparations provide a method to study the overall structural and functional properties of membrane proteins. However, organelle specific information is not available in this approach. Alternatively, separating vesicles based on their original organelle could provide information on the assembly and trafficking of membrane proteins. For example, it has been hypothesized that nicotine actsas a pharmacological chaperone of $\alpha4\beta2$ nicotinic receptors and nicotine alters the assembly of the nicotinic receptors towards the high sensitivity isoform in the ER. To validate this hypothesis, we isolated $\alpha4\beta2$ nicotinic receptors located on vesicles derived from the ER and plasma membrane origins and utilized single molecule studies to determine the stoichiometric assembly of the receptor. The data suggested that the ER has a higher ratio

of the low sensitivity isoform ($(\alpha4)_3(\beta2)_2$) than the plasma membrane indicating that the high sensitivity isoform trafficked more efficiently to the cell surface. When nicotine was added, the distribution of nicotinic receptors changes in those compartments. In both the ER and plasma membrane, the percentage of high sensitivity isoform was greater than the sample without the presence of nicotine. The results suggested that nicotine altered the assembly of nicotinic receptors to form the high sensitivity isoform in the ER and the altered assembly trafficked to the plasma membrane efficiently increasing the ratio of this isoform in the plasma membrane.

The cell derived vesicles we utilized to isolate single receptors are structurally similar to liposomes, an FDA approved drug delivery system, which is spherical vesicles composed of at least one lipid bilayer. Hence, cell-derived vesicles possess potential to be utilized as drug delivery vehicles. I explored the applicability of cell-derived vesicles as general delivery vehicles to cultured cells. Additionally, we implanted xenografts into immune compromised nude mice and prepared cell derived vesicles labeled with dye molecules. The vesicles were injected in a mouse containing a xenograft to monitor whether these vesicles can reach to the xenograft. Our data suggested that cell-derived vesicles can successfully reach the xenograft and thus have potential to be utilized as a drug delivery vehicle.

KEYWORDS: single molecule, nicotinic receptors, stoichiometry, drug delivery, vesicles

Faruk Hossain Moonschi

05/13/2018

APPLICATIONS OF CELL-DERIVED VESICLES: FROM SINGLE MOLECULE STUDIES TO DRUG DELIVERY

By

Faruk Hossain Moonschi

Dr. Christopher I. Richards

Director of Dissertation

Dr. Mark A. Lovell

Director of Graduate Studies

05/13/2018

Date

To My Parents

Acknowledgements

First and foremost, I would like to thank Dr. Chris Richards for accepting me in his group and giving me the opportunity to work in his lab. Without his constant guidance and support, I would not be able to finish my doctoral training. I would also like to acknowledge his hands-on training in molecular biology, tissue culture, fluorescent microscopy and Matlab.

I would also like to thank all members of my graduate committee–Professors Anne-Frances Miller, Bert C. Lynn, Chang-Guo Zhan– for their constant support and guidance. I would like to thank my outside examiner, Professor H. Pete Spielmann, for serving in my defense committee.

I want to thank my collaborators in the research groups of Professors Jim Pauly, Jill Kolesar, Harley Kurata, Pete Kekenes-Huskey, Jason DeRouchey, Vince Venditto, Brad Berron and Edith Glazer. I am grateful to Deann Hopkins for helping me to learn mice surgery, cardiac perfusion and brain extraction, Dr. Rob McCorkle for xenograft implantation and vesicle delivery studies in mice, Dr. Dave Heidary for providing me with ideas to conduct experiments on various occasions and allowing me to use instruments in his laboratory.

I want to thank all current and past members of Dr. Richards' group for helping me with various laboratory techniques. I would like to especially thank Aaron Snell for reviewing my dissertation and giving me feedback and Dr. Ashley Loe for collaborating with me in various projects.

I thank all my past and present teachers. Without their help, I would not be able to come this far in my career. I would like to thank all members of the chemistry business office for your constant help and support.

I thank all of my family members for their constant support, encouragement and unconditional love. Most importantly, I would like to thank my wife, Dr. Minakshi Bhardwaj, for her love, kindness and support. Without her constant love and care, it would have been difficult to finish my dissertation.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ACh, Acetylcholine

AD, Alzheimer Disease

Bup, Bupropion

CCD, Charged-Couple Device

COPI, Coat Protein I,

COPII, Coat Protein II,

CFTR, Cystic Fibrosis Transmembrane Conductance Regulator

CFTR-GFP, CFTR conjugated with green fluorescent protein

Cyt , (—)-Cytisine

DDS, Drug Delivery System

EC_{50} , Half Maximal Effective Concentration

ECS, Extracellular Solution

EMCCD, Electron Multiplying Charge-Coupled Device

EGF, Epidermal Growth Factor

EGFR, Epidermal Growth Factor Receptor

ER, Endoplasmic Reticulum

FCS, Fluorescence Correlation Spectroscopy

FRET, Forster Resonance Energy Transfer

GBD, Glass Bottom Dish

GFP, Green Fluorescent Protein

HEK293T, Human Embryonic Kidney 293T Cell Line

IVIS, *In Vitro* Imaging System

ID, Integrated Density

LMV, Large Multilamellar Vesicles

mCherry, a monomeric red fluorescent protein named as mCherry

nAChR, Nicotinic Acetyl Choline Receptor

MVB, Multi-Vesicular Body

Nic, (—)-Nicotine

PBS, Phosphate Buffered Saline

PM-mCherry, Plasma Membrane Cherry

SEM , Standard Error of the Mean

SEP, Super-Ecliptic PHluorin

SUV, Small Unilamellar Vesicle

TIRF, Total Internal Refection Fluorescence

TMR, Tetramethyl Rhodamine

PD, Parkinson's Disease

PM, Plasma Membrane

PMID, Plasma Membrane Integrated Density

RT, Room Temperature

ROI, Region of Interest

Var, Varenicline

wt, Wild Type

Chapter 1

Introduction to Single Molecule Studies and Drug Delivery Systems

1.1 Aims and Scopes of This Study

One of the primary aims of this work was to develop a method to isolate single membrane proteins into a physiological relevant lipid bilayer to conduct single molecule studies. One of the biggest challenges for single molecule studies of membrane proteins is the concentration barrier resulting from high levels of proteins present in the native cellular environment. The approaches detailed in this work provide the ability to isolate single proteins in their physiological environment without compromising the structural integrity of multimeric proteins.

Current approaches to studying membrane receptors often rely on biochemical techniques that are unable to distinguish between receptors originating from different organelles. Another aim of this work was to utilize cell derived vesicles to determine the overall structure and assembly of a membrane protein from specific organelles. This approach provided us with the capability to study the properties of multimeric proteins originating from the endoplasmic reticulum and the plasma membrane in order to understand their assembly before and after trafficking to the plasma membrane.

The final aim of my work was to utilize cell-derived vesicles to deliver therapeutics into a targeted location. Lipid-based drug delivery vehicles including liposomes and exosomes have been extensively studied to encapsulate therapeutics and to deliver them into a targeted location in the body (1-3). Cell-derived vesicles those used to isolate single receptors have many properties that make them ideal for the encapsulation and delivery of

therapeutics. We validated that cell-derived vesicles can be used to deliver different types of cargo both to cells and to tumors in a rodent model.

In Chapter 1, I describe background information related to my research including (1) the challenges faced by researches who conduct single-molecule based studies of biological systems, (2) the different approaches made by researchers to isolate single molecules, (3) the different single molecule methods used in this work, (4) the membrane proteins used in this study, and (5) the common approaches used by researchers to utilize lipid based systems as a drug delivery system (DDS) and advantages of using cell-derived vesicles as a DDS.

In Chapter 2, I describe a single molecule method which isolates single receptors into cell-derived vesicles. I utilized HEK293T cells which expressed α3-GFP β4-wt or α3-wt β4-GFP receptors. Nitrogen cavitation and differential ultracentrifugation were employed to isolate cell-derived vesicles implanted with single receptors. These vesicles were utilized to study the stoichiometry of the receptor at the single molecule level. The data suggested that $(\alpha3)_2(\beta4)_3$ is the predominate stoichiometry of the receptor. We verified the result using two color experiments with α3-mCherry β4-GFP receptors. I also demonstrated that cell-derived vesicles can be employed to study ligand receptor interactions using fluorescence correlation spectroscopy (FCS) and immobilized molecules on a glass substrate.

In Chapter 3, related to the second aim of this work, I included the studies showing the effect of different types of nicotinic receptor ligands (agonists, partial agonists and

antagonist) on the assembly and trafficking of α4β2 nicotinic receptors. Then, ER and plasma membrane originated α4β2 nicotinic receptors were isolated to study the stoichiometric assembly of the receptor in those organelles. The data suggested that all types of nicotinic ligands increased the total number of receptors in the ER and the plasma membrane and altered the distribution of receptors in those organelles. The single molecule studies with whole cell samples suggested that all nicotinic ligands differentially shifted the stoichiometry from the low sensitivity isoform ($(\alpha4)_3(\beta2)_2$) to the high sensitivity isoform ($(\alpha4)_2(\beta2)_3$). The single molecule studies with the ER and plasma membrane originated vesicles suggested that (–)-nicotine (hereafter, nicotine) altered the assembly of nicotinic receptors in the ER and the high sensitivity isoform is preferentially trafficked from the ER to the plasma membrane.

In Chapter 4, I included work related to the utilization of cell-derived vesicles as a delivery vehicle *in vitro* and *in vivo*. These studies utilized fluorescent dyes and separately chemotherapeutics for delivery to cells to demonstrate their capability as a drug delivery vehicle. We characterized the vesicles, implanted xenografts, collected and purified cancer cells, and injected the cancer-cell-derived vesicles into mouse models to monitor if vesicles can reach to the tumor. Our preliminary data suggested that cancer cell-derived vesicles can reach to the tumor selectively, but will need to be optimized to facilitate a robust delivery of cargo into the tumor.

1.2 Challenges of Single Molecule Studies in Biological Systems

Biological systems are very complex because of the thousands of processes occurring simultaneously. A better understanding of these processes can provide critical information for the development of cures to treat diseases. A common approach to understand biological processes involves ensemble studies which can provide valuable information about the system but can lose some other information as well. For example, calcium sensing fluorescent dyes can be loaded into cells and the calcium induced fluorescence signal can be monitored after activating calcium channels. This result can provide information relating the amount of ligand added versus the degree of activation of ion channel. However, the activity of individual calcium channels is lost due to averaging over the entire ensemble. A single-molecule analysis of the channel can provide information of channel activation, conformational changes, and gating dynamics (4-6). Understanding these processes can provide key insights needed to design better pharmaceuticals to manipulate the ion channel.

Biological phenomena including protein conformational changes, protein folding and unfolding, ion channel gating, and oligomeric protein assembly have been studied with single-molecule fluorescence microscopy (4,6-12). For example, single molecule Forester resonance energy transfer (FRET) can be utilized to determine the distance between specifically labeled dye molecules on two different positions of a biomolecule. The change in distance can be utilized to determine the conformation change and the gating mechanism of ion channels. Folding and unfolding of a protein can also be studied with single molecule

FRET (11,12). Single molecule photobleaching step counting can provide information about the stoichiometric assembly of an oligomeric protein (10,13).

One of the challenges of single molecule studies of biological samples with fluorescence microscopy is that one molecule needs to be isolated in a relatively large area defined by the diffraction limit of light. A diffraction limited spot is a theoretical limitation of an optical lens due to the diffraction of light. If two fluorophores are located in a distance less than or equal to the diffraction limit of the objective lens, their signal will display a single diffraction limited fluorescent spot rather than two separate spots (Figure 1.1). The size of a diffraction limited spot is determined by Abbe resolution, which, for microscopy, can be expressed as,

$$\boldsymbol{Abbe\ resolution} = \frac{\boldsymbol{\lambda}}{\boldsymbol{2NA}} \qquad \textbf{Eq. 1.1}$$

Where λ is the wavelength of the light and NA is the numerical aperture of the objective lens. NA of an objective lens is the range of angle to which it can emit light or from which it can accept light. Mathematically, NA can be expressed with the following equation,

$$\boldsymbol{NA = \eta \times \sin(\theta)} \qquad \textbf{Eq. 1.2}$$

Where η is the refractive index of the media and θ is one-half of the maximum angle of the objective lens.

The native concentration of proteins expressed in a cell is usually not suitable for single molecule studies due to the presence of multiple proteins in a diffraction limited

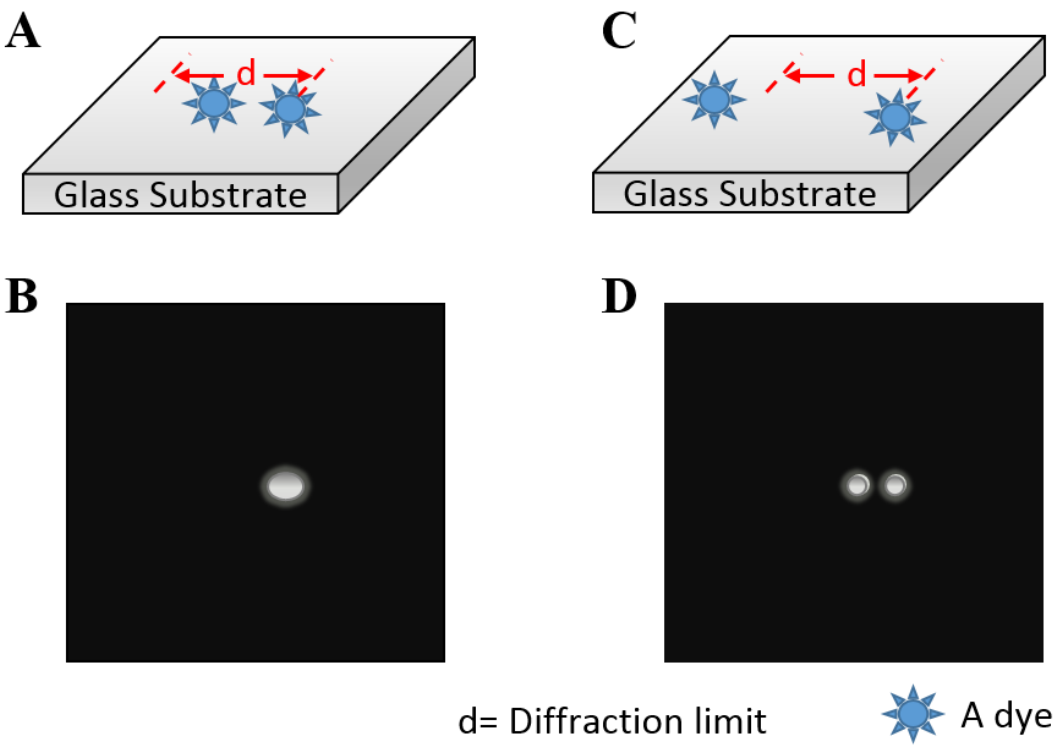

Figure 1.1 Graphical representation of the consequence of the presence of two fluorophores in a distance lower than or equal to the diffraction limit of a microscope. (A) Two fluorophores are located at a distance lower than the diffraction limit of the objective lens. (B) When an image of the fluorophores in "A" is taken with a camera, the image will display a single spot due to the inability of the objective lens to separate the emissions from the two fluorophores. (C) When two fluorophores are separated by a distance more than the diffraction limit, (D) the image will provide two distinct spots corresponding to the two fluorophores.

spot. One approach researcher usually take to mitigate this challenge includes the expression of protein at a very low concentration which can be achieved by transfecting the cells with fewer plasmids and reduced time (13,14). However, cells generate background fluorescence from a number of sources including NADH, riboflavin, flavin coenzymes (15) as well as mitochondria (16), and this noise reduces the signal to noise ratio.

Additionally, membrane proteins tend to move laterally along the membrane (17) spreading the emitted photons across multiple pixels of the camera. This process limits single molecule studies to the few applications that can take advantage of bright fluorophores such as quantum dots. Another approach of isolating single proteins utilizes artificial lipid bilayers (18-21). The general approach requires the expression of proteins in a cellular system with subsequent isolation and purification of the protein of interest. This method works well for soluble proteins but is more challenging for membrane proteins because they need to be transferred into an artificial lipid bilayer. One issue with this approach is that the protein needs to be temporarily solubilized outside of a membrane in a detergent solution. The detergent solution can denature and sometime precipitate the proteins (22,23). This can lead to an oligomeric protein or a protein with multiple subunits to fall apart and to lose functional activity. Another issue with this approach is that artificial lipid bilayers lack the same membrane components as the native cellular membrane. Hence, a better approach is necessary which can isolate single oligomeric proteins into their physiological membrane. Additionally, to study single molecules using fluorescent tags, a fluorescence microscope needs to be selected which can enhance signal to noise ratio and possesses ability of detecting signals from single fluorophores.

1.3 Fluorescence Microscopy

Fluorescence microscopy is an integral part of the study of many biological systems both at the single cell and single molecule level. Microscopy can be used to visualize fluorescent proteins or organic dyes which are selectively incorporated into a specific part of a biological system. For example, lipids, proteins or DNA of a cell can be labeled with fluorescent probes which can be excited with a light source to visualize the labeled molecules using a fluorescence microscope. Thus, a cell or its compartments can be visualized and studied. Different types of fluorescence microscopy have been developed and are commercially available. A brief description of the most common ones are as follows:

1.3.1 Wide-Field Fluorescence Microscopy

In this microscopy technique, an excitation source is used to illuminate all parts of a specimen and the resultant fluorescence emission is either visualized by eye or captured as an image with a camera (Figure 1.2). A xenon or mercury lamp is used as an excitation source which is passed through a filter to select a narrow band of light and is directed to the specimen through an objective lens. The emission is passed through a dichroic mirror and/or a filter to separate the emission from the excitation source, and the emission is finally directed to a camera to capture image of the specimen. Since the whole sample is illuminated and emission is generated from the entire specimen, the presence of out of focus emission reduces the image contrast and resolution. One of the common approaches to resolve this issue involves the use of very thin specimens (less than 10 μm) which reduce the emission signal and thus improves the image contrast (24). The major advantages of

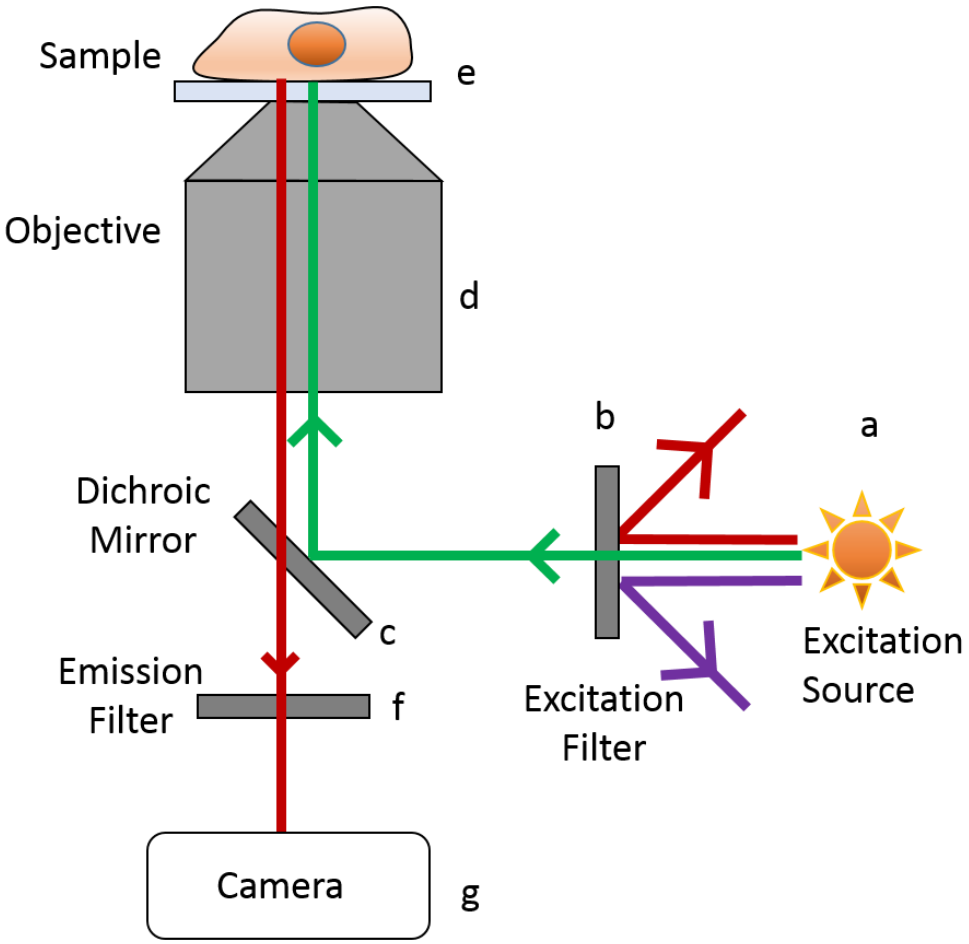

Figure 1.2 A schematic representation of the different parts of a wide-field microscope. Excitation source is usually generated from a lamp (a) which is directed to an excitation filter (b). This filter usually passes a narrow band of wavelengths while reflecting all other wavelengths. The selected band from excitation source is directed to a dichroic mirror (c) which reflects the light toward the sample (e) through an objective (d). The fluorescent molecules located in the sample become excited by the excitation source and emit fluorescence. A part of the emission passes though the objective toward the dichroic mirror which allows the emission to pass through. The emission is further filtered using an emission filter (f) and directed to a camera for capturing an image of the specimen.

wide-field fluorescence microscopy include low cost, simple instrumentation, and flexibility. The major drawback of this system arises from the technique of exciting the entire specimen which causes photobleaching of the incorporated fluorescent probes and photo-toxicity to the live cell under investigation. The other notable disadvantage is the low resolution of the images.

1.3.2 Confocal Microscopy

In wide field microscopy technique, the emission originated from the out of focus molecules causes blurriness in the image of a specimen. Confocal microscopy is a special type of fluorescence microscopy which provides better image resolution. In this microscopy technique, the excitation source is usually a laser which is brighter and smaller in size than that obtained from the mercury lamp used in wide-field microscopy. The excitation lights from the laser is usually directed towards a sample through a pinhole and an objective to produce a confocal beam, and the emitted lights are collected through the same objective. A pinhole aperture is placed on the path of the emitted light to exclude all light except that generated from the fluorophores located at the focus of the confocal beam. Thus, the out of focus emission cannot reach to the detector providing an image of better resolution. The emission is usually detected by a photomultiplier tube (PMT) for every single position on the sample or the region of interest. Therefore, either the stage holding the sample on top of the objective lens needs to be moved in different positions in space (25) or the laser source needs to be directed toward different parts of the specimen (26) in order to record an image. In the former arrangement, usually the stage is physically moved along the X and Y axis while in the later arrangement, the laser is directed toward different

parts of the specimen by using two orthogonal-mirrors (24). The laser and pinhole apertures stay unchanged in their positions in both types of set-ups. Laser scanning confocal microscopy is much faster and thus more common.

1.3.3 Total Internal Reflection Fluorescence Microscopy

A confocal microscope can acquire images of better resolution than a traditional a wide-field fluorescence microscope and can produce an image from optical sections of about 1 μm. In contrast, total internal reflection fluorescence (TIRF) microscopy can produce optical sections of about 100 nm. TIRF microscopy is a special type of fluorescence microscopy which is used to illuminate a thin segment of a sample located above a glass substrate. In this microscopy technique, the excitation light is directed through a high refractive index medium (usually glass substrate) into a low refractive index medium (solution) at an angle. When the incident angle increases, the refraction angle also increases. When the incident angle becomes larger than a critical value, the light does not transfer from high refractive medium to low refractive medium. Rather, the excitation source undergoes total internal reflection producing an exponentially decaying light from the interface of the two media into the low refractive index medium. This exponentially decaying light is known as evanescent wave and has the same wavelength as the excitation source. The evanescent wave is utilized to excite the fluorescent molecules mounted on a glass substrate. The penetration depth of the evanescent wave depends on the wavelength of excitation source, the refractive index of the media and the incident angle utilized to obtain total internal reflection (24). Since a TIRF microscope excites only a narrow region (100-200 nm) into the sample, the background fluorescence decreases. It was reported that

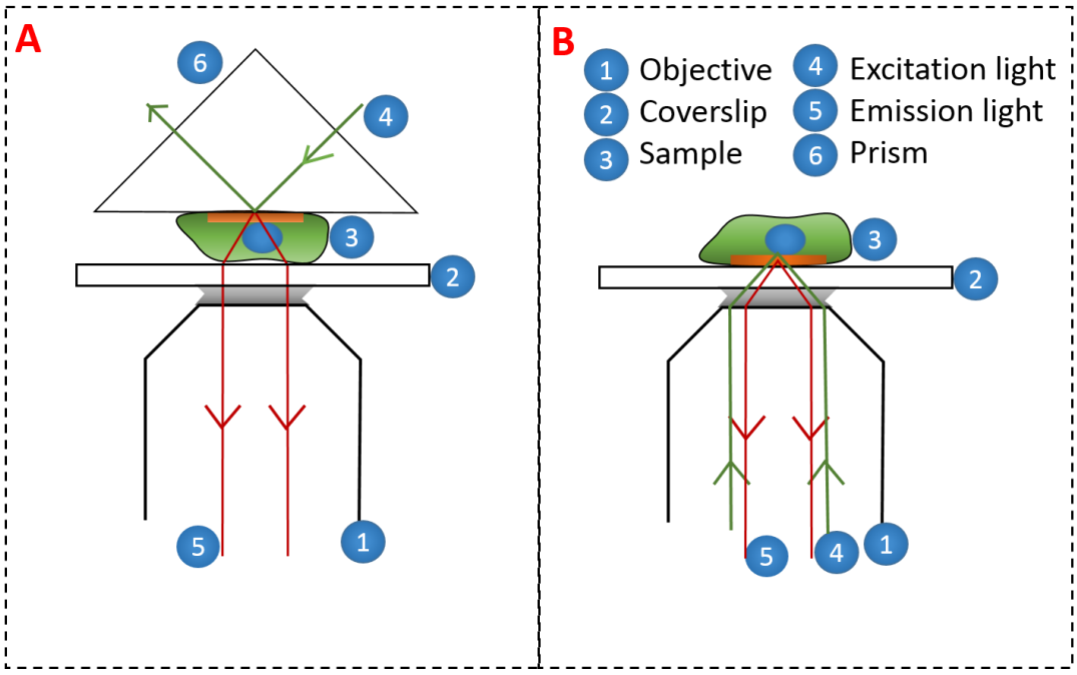

Figure 1.3 A schematic representation of a prism-type and an objective-type total internal reflection fluorescence (TIRF) microscope set-up. (A) In a prism-type TIRF microscopy set-up, an excitation beam is directed through a prism to the sample present on top of a coverslip, and the emission is collected through an objective present under the coverslip. (B) In an objective type TIRF microscopy set-up, the excitation beam is directed into the sample located on top of the objective and the emission is collected through the same objective.

background fluorescence is 2,000 time lower in a TIRF microscope image than that obtained in a typical wide-field fluorescence microscopy (27).

Single molecule studies of immobilized molecules are usually conducted with TIRF microscopy equipped with an objective of a very high magnification (60-120x) and a camera capable of detecting very low signals (10,28,29). An electron multiplying CCD camera can be utilized to collect images of single molecules. The cooling system within the camera reduces shot noise.

There are two common types of set-ups available for TIRF microscopy – Objective type, and Prism type (30). In prism type TIRF microscopy, a prism is placed on top of a sample present on a coverslip, and an inverted objective is positioned on the bottom of the coverslip (Figure 1.3A). The excitation light is directed through the prism to the sample, and the emission light is collected through the objective. For this system, a sample is usually prepared in a microfluidic device whose top and bottom are made of a glass slide and a cover slip respectively (31). This device with the sample is mounted on top of the objective and the prism is mounted on top of the sample. The sample is excited from the top since the excitation signal passes through the prism, and the fluorescence signal is collected from the other side of the device though the objective.

In an objective type TIRF microscope, the sample is placed on top of a coverslip which is mounted on top of an inverted objective of high numerical aperture (>1.45)(Figure 1.3B) (29,32). To achieve total internal reflection, the excitation beam is focused at the back aperture of the objective. Then, a stepper motor is used to laterally move the beam

toward the edge of the objective. This approach increases the angle of the incident light at the interface of the glass substrate and the sample. When the objective has a high enough numerical aperture, the excitation beam can be directed at an incident angle greater than the critical angle needed to produce a total internal reflection of the excitation source. The fluorescence emission is collected through the same objective and is separated from the excitation light using a dichroic mirror.

1.4 Novel Approaches in Single Molecule Studies

To conduct single molecule studies of biological molecules, a single molecule needs to be isolated into a diffraction limited spot. The normal expression of membrane proteins in cells is often not suitable to conduct single molecule studies. To overcome this concentration barrier, researchers have employed a number of approaches including encapsulation of proteins into an artificial membrane (18,19), employing a metal nano-aperture (10) or simply expressing the protein at a very low concentration (14). However, each of these approaches has limitations as presented previously (section 1.2). Here we utilize a novel approach where cell derived vesicles are employed to isolate single receptors.

1.4.1 Cell-Derived Vesicles

Cell derived vesicles can be generated by homogenizing cells obtaining fractions of a cell inserted with membrane proteins. Common approaches to prepare the vesicles includes ultrasonic and other mechanical homogenization as well as nitrogen cavitation (33,34). All these approaches are assumed to produce cell fragments containing a native

lipid bilayer which spontaneously form spherical vesicles. The mechanism of the spontaneous formation of spherical vesicles from lipid bilayers has been explained based on thermodynamics (35) and validated using molecular dynamic simulations (36). The basic mechanism involves the structural features of a lipid molecule which contain a hydrophobic tail and a hydrophilic head group. The hydrophobic tails of lipid molecules do not prefer to be associated with water molecules due to hydrophobic interaction. Rather, the hydrophobic tails interact with each other via Van der Waals interactions. The process of self-association of the hydrophobic tails is thermodynamically favorable over the interaction of the hydrophobic tails with water molecules (35,36). The minimum interaction of hydrophobic tails of a cell fragment with water molecules is achieved when the ends of a cell fragment associate with each other forming a spherical vesicle. In this process, the entropy of the water increases making the process of vesicle formation thermodynamically favorable (36).

Another common method of vesicle formation is extrusion which is typically used to decrease the size of liposomal vesicles (37,38). However, this method has also been applied to prepare cell-derived vesicles from mammalian cell suspension. In this approach, a cell slurry is usually forced through a series of polycarbonate membranes containing very small pores to generate vesicles inserted with endogenous proteins (39,40). To explain the mechanism of vesicles formation via extrusion, Clerc et al has proposed a mechanism based on the study of soap bubbles (41). According to this theory, a pore in a polycarbonate membrane can be considered as a cylindrical tube, and when a large vesicle is passed through the tube, a cylindrical lipid bilayer is formed inside the tube. When this cylindrical structure reaches the other side of the pore, the cylindrical lipid bilayer breaks and

reorganizes to produce small vesicles. This theory is based on the experiments carried out with soap bubbles which showed that at the end of the pore, the bubble breaks when the length of the bubble reaches to a critical value equal to the circumference of the pore.

Spherical vesicles generated by homogenization of cells can vary in size and may contain multiple transmembrane proteins. The size of these vesicles can be reduced by passing them through a polycarbonate membrane used in extrusion. Although, different types of homogenization can produce cell derived vesicles, nitrogen cavitation has several advantages (33) over other methods including (i) no heat damage because of the lack of shear stress and friction compared to lysis, (ii) minimal physical and chemical stress compared to mechanical homogenization, (iii) reduction in temperature during sudden release of the pressure (the adiabatic expansion), (iv) a better protection of the labile cell compartments as nitrogen does not cause any oxidation of the compartments, (v) no change in the pH of the solution, (vi) a fast and uniform process, and (vii) commercial availability of instruments for variable sample sizes.

1.4.2 Isolation of ER and Plasma Membrane Specific Vesicles

Cell derived vesicles generated from whole cell lysate contain vesicles originated from both the ER and the plasma membrane. These vesicles can be employed to study the inserted proteins using single molecule techniques and can provide an overall picture of the whole cell. However, the organelle specific information of the membrane proteins is lost but the information can be obtained by isolating cell-derived vesicles originating from the different organelles.

To isolate the freshly synthesized receptors in the ER and the resident receptors in the plasma membrane, we aim to fragment the membrane using nitrogen cavitation and to isolate the vesicles based on their endogenous density difference. The ER of mammalian cells contains low concentrations of cholesterol and sphingolipids which impart flexibility to the ER membrane allowing it to form the flattened sacs-like structure of the membrane (42). The plasma membrane of mammalian cells is composed of a higher percentage of cholesterol and sphingolipids compared to that in the ER allowing for a compact assembly of the lipids in the plasma membrane. Therefore, the density of the lipids becomes higher in the plasma membrane than that in the ER and thus the plasma membrane is resistant to external mechanical stress (42). However, the rough ER which houses the proteins under investigation contains a plethora of ribosomes which makes the membrane of the rough ER much denser than the plasma membrane housing transmembrane proteins for normal cellular functions (43). Hence, a density gradient can be employed to isolate ER and plasma membrane derived vesicles.

The most common method of isolating intracellular organelles or subcellular vesicles is the application of sucrose gradient. This approach involves the placement of the vesicle solution on top of the layers of sucrose of different densities and the centrifugation of the sample to help the vesicle solution to penetrate into the sucrose gradient. The separation process is dictated by Stoke's law: $V_t=2R^2a(\rho_p - \rho_f)/(9\eta)$, where V_t is the terminal velocity of the particles falling throgh the gradient, R is the radius of the particle, a is the applied acceleration force applied by the centrifugation, ρ_p is the density of the particles, ρ_f is the density of the fluid (sucrose gradient) and η is the viscosity of the medium (44). When the terminal velocity of the vesicles is zero due to the resistance from the

sucrose solution, the organelles do not penetrate further inside the solution. The terminal velocity of the vesicles inside the solution depends on the density and size of the penetrant (44). The particles with larger radius and higher density can penetrate deeper into the gradient solution and settle into the higher density fraction.

However, one of the major disadvantages of using a sucrose gradient (45) is that it cannot maintain isosmotic pressure in the different density gradients. The vesicles or organelles being isolated suffer from hyperosmotic pressure lightly in the low density regions and severely in the highest density fractions of the sucrose solution. Thus, the vesicles continuously lose water to the medium during penetration through the sucrose gradient solution. This process endangers the hydrated macromolecules of the vesicles.

Conversely, OptiPrep is an iodinated density gradient solution with 60% iodixanol in water and can maintain isosmotic pressure up to the density of 1.32 g/ml corresponding to the 60% (w/v) solution (45). This solution is nonreactive to the biological materials, does not remove water from the vesicles allowing them to maintain their size, and has been employed to isolate subcellular vesicles and organelles (46,47).

1.4.3 Step-wise Photobleaching of Single Fluorophores

A total internal reflection fluorescence microscope can help us to visualize single fluorophores by exciting about 100-200 nm into the sample above the glass substrate and by increasing the signal to noise ratio. However, under continuous excitation, fluorophores ultimately undergo an irreversible transition into a dark state. This process is known as the

photobleaching of the molecule. Photobleaching is usually considered undesirable and a limitation of fluorescence microscopy.

The mechanism of photobleaching is primarily believed to be the result of a chemical reaction of the fluorophore with the reactive molecules in the local environment. Fluorescent molecules transition to an excited singlet state and these molecules normally go back to the ground electronic state from the first excited electronic state releasing the energy as photons. But sometimes an excited molecule transitions from the excited singlet state to an excited triplet state. Molecules are highly energetic in the excited triplet state and can react with the ground electronic state of molecular oxygen. This process not only helps the fluorescent molecules to return from the temporary dark state (triplet state) to ground electronic state but also generates highly reactive excited singlet oxygen molecules. An excited singlet oxygen molecule can react with the exposed functional groups of organic dye molecules (e.g. amines) and exposed amino acid side chains of fluorescent proteins (e.g. cysteine, histidine, tryptophan, and tyrosine) (48,49). The reaction between exposed groups of dye molecules and excited singlet oxygen molecules changes the structure of dye molecules transferring them to a permanent dark state, called photobleaching. The reaction between excited singlet oxygen and dye molecules depends on (i) the excitation intensity, (ii) the concentration of molecular oxygen, (iii) temperature, and (iv) the rigidity of the medium (50). Jouonang *et. al.* have shown that photobleaching survival time probability distribution, by measuring thousands of molecules, follows a monoexponential decay curve (50). Hence, photobleaching survival time (simply photobleaching time) of dye molecules is a stochastic event. A number of approaches have

been used to reduce photobleaching of molecules including enzymatic reactions to remove the oxygen from the solution (51,52).

However, careful study of the photobleaching nature of single molecules has provided an opportunity to understand the assembly of biological complexes (10,13,28). When temperature, buffer and the nature of fluorescent molecules are constant, the photobleaching time is a stochastic event and is dependent on the intensity of the excitation source. At a low excitation intensity, the probability of photobleaching of two fluorophores at a given time is minimal. Hence, a low excitation intensity allows different fluorophores to bleach at different time points. As the total fluorescence intensity arises from all the fluorescent molecules present, the fluorescence intensity is reduced by one step when a fluorophore is photobleached. Therefore, the number of photobleaching steps indicates the number of fluorescent molecules present. If a subunit of a multimeric protein is genetically tagged with a fluorescent protein, counting the number of photobleaching steps in the expressed protein can provide the number of fluorophores present in the protein. Hence, the number of subunits presents in a multimeric protein can be determined as well.

1.4.4 Fluorescence Correlation Spectroscopy

Fluorescence correlation spectroscopy (FCS) is another single molecule fluorescence intensity based technique and can be employed to analyze biomolecules in a very low concentration (about 1 μM to 1 nM) with high spatial and temporal resolution (53). When a fluorescent molecule diffuses through a confocal laser beam, the fluorescence intensity of the molecule spontaneously fluctuates (54-56). This property of fluorescence

intensity fluctuation is utilized in FCS to determine the parameters responsible for the fluctuation (53).

In this method, a fluorescent molecule is allowed to diffuse through a confocal beam to produce fluorescence signal and the fluorescence intensity obtained from the focal plane of the confocal beam is recorded over time. Then, an autocorrelation function of the signal is determined. In principle, the autocorrelation function, G(τ), is a measurement of the similarity of the signal with itself after a certain lag time (τ) and can be express as follows:

$$\mathbf{G(\tau)} = \frac{\langle \delta F(t) \delta F(t+\tau) \rangle}{\langle F(t) \rangle^2} \qquad \textbf{Eq. 1.3}$$

$$= \frac{\langle F(t) F(t+\tau) \rangle}{\langle F(t) \rangle^2} - \mathbf{1} \qquad \textbf{Eq. 1.4}$$

Where, F(t) = fluorescence intensity at time t

F(t + τ) = fluorescence intensity at time t + τ

δF (t) = F(t) - <F(t)>

δF (t + τ) = F(t + τ) - <F(t)>

The y-intercept of the normalized autocorrelation function, G(0), can provide information of the average number of molecules (<N>) present in the focal volume. The average number of molecules in the focal volume can be obtained with following equation:

$$\langle \mathbf{N} \rangle = \frac{\mathbf{1}}{\mathbf{G(0)}} = \frac{\mathbf{1}}{\mathbf{V_{eff}} \langle \mathbf{C} \rangle} \qquad \textbf{Eq. 1.5}$$

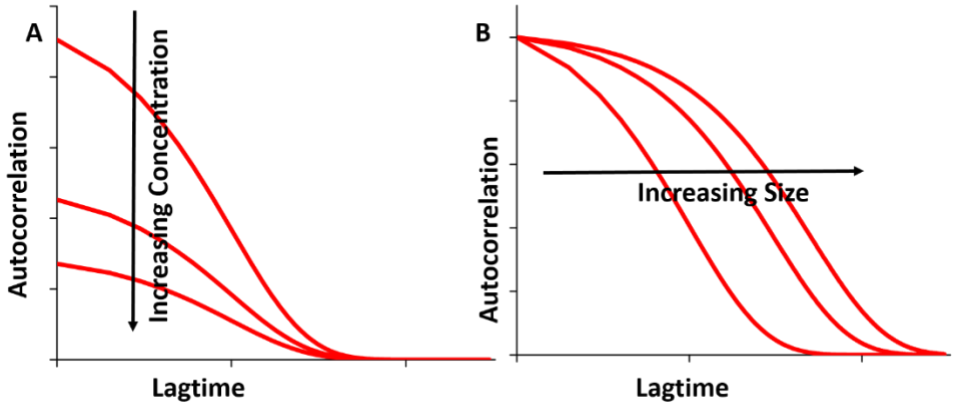

Figure 1.4 Typical autocorrelation curves obtained from fluorescence correlation spectroscopy (FCS). (A) The y-intercept of an autocorrelation curve inversely represents the number of molecules present in the focal volume. Thus, with the increase in the concentration of molecules, the y-intercept decreases. (B) The half-value decay time is a good approximation of diffusion time which is related with the size of the molecule. Hence, as the size of the molecule increases, the diffusion time also increases.

Where, <C> = average concentration

V_{eff}= Effective focal volume

Hence, if the concentration of molecules increases, the y-intercept of the autocorrelation function G(0) decreases (Figure 1.4A). The half-value decay time provides a good estimate of the average diffusion time, and when the diffusion time increases due to the increase in size of the molecule, the half value decay time also increases (Figure 1.4B). The diffusion time (τ_D) can be used to determine the diffusion coefficient (D) of the fluorescent molecule using the following equation:

$$\textbf{Diffusion coefficient, D} = \frac{W_{xy}^2}{4\tau_D} \qquad \textbf{Eq. 1.6}$$

Where, W_{xy} is the radial radius of the confocal beam and τ_D is the diffusion time. The diffusion coefficient can be employed to deduce the size of vesicles using Stokes–Einstein equation. The Stokes–Einstein equation can be expressed as:

$$\textbf{Radius of the particle, r} = \frac{k_B T}{6\pi\eta D} \qquad \textbf{Eq. 1.7}$$

Where, k_B is the Boltzmann constant, T is the temperature in Kelvin, η is the viscosity of the solution, and D is the diffusion coefficient.

1.5 Nicotinic Acetylcholine Receptors

I have conducted single molecules studies with nicotinic acetylcholine receptors (nAChRs) because of their physiological importance and their association with different diseases and nicotine addiction. nAChRs are ligand gated ion channels belong to the Cys-

loop super family of proteins. These proteins are expressed throughout the central and peripheral nervous systems and can be activated by both acetylcholine, an endogenous agonist, and nicotine, a compound found in tobacco (57). Nicotinic receptors can be divided into muscle-type and neuronal-type. Muscle-type nAChRs are found in the neuromuscular junction while neuronal-type nAChRs are expressed throughout the central and peripheral nervous system. These receptors are very important to normal physiological function but are also associated with different diseases.

1.5.1 Physiology and Pathology of Nicotinic Receptors

Nicotinic receptors are located both in the presynaptic and postsynaptic junction (58). In postsynaptic regions, these receptors upon binding with the endogenous neurotransmitter, acetylcholine, lead to the depolarization of the neuron which initiates action potentials (59,60). In presynaptic regions, these receptors are believed to affect their local environment and thus help to release neurotransmitters into the synaptic cleft (61) This is done by the flow of sodium ions into the cells activating voltage gated calcium channels which helps the neurotransmitter containing vesicles to fuse to the cell membrane releasing the cargo into the synaptic cleft (62). Some researchers believe that a nicotinic receptor's ability to flux calcium ions into the cell is sufficient to elicit the release of the neurotransmitter (63-65).

Nicotinic receptors located in the brain are believed to be involved in learning and memory formation, neuronal development, consciousness and reward (59,66). Since these receptors plays a very important physiological role, dysfunction in the channel opening and

closing or their relative expression level in the brain can be involved with diseases. For example, Parkinson's disease (PD), and Alzheimer disease (AD are the most common diseases which has been associated with low level of nicotinic receptors (67-72). Postmortem brain analysis of AD and PD patients exhibited a significant decrease in the quantity of nicotinic receptors in the frontal cortex, temporal cortex, hippocampus and caudate nucleus (67). A similar study with the brains of patients suffering from Schizophrenia displayed a significant decrease in nicotinic receptors in the hippocampus (73). These receptors are involved in nicotine dependency as well.

1.5.2 Structure of Nicotinic Receptors

In the brain, nine alpha subunits (α2-α10) and three beta subunits (β2-β4) of nAChRs have been isolated. Some of the common structural properties of a nAChR subunit include the following (Figure 1.5) (57,74,75):

i. Four transmembrane domains (M1-M4) where M2 forms the line of the central pore and M4 is located towards the outside of the receptor,
ii. A long extracellular N-terminal domain providing glycosylation sites consisting of ten beta stands (β1- β10) and two alpha helices (α1- α2).
iii. A conserved disulfide bond formed between two cysteines of 13 amino acids apart and by joining the beta stands β6 and β7
iv. A large intracellular domain between transmembrane domains M3 and M4 presenting phosphorylation sites,
v. A short C-terminal domain (4-28 amino acids residues),

Five subunits of nAChRs arrange symmetrically to generate a central ion pore which non-selectively transports cations (Figure 1.5B). Heteromeric nAChRs consist of two or more different types of subunits; homomeric ones are composed of all the same subunits (α7-α9). The two main nAChRs found in the brain are α4β2 and α7 where these receptors are heteromeric and homomeric, respectively (Figure 1.5C and D). Each heteromeric nAChR contains at least two agonist binding sites that are located on the N-terminal domain between an alpha and a non-alpha subunit (57,74). On the other hand, homomeric nAChRs contain five agonist binding sites (76). In heteromeric nAChRs, alpha subunits containing a cysteine doublet make the positive side, but the non-alpha subunits make the negative side of the agonist binding pocket. In contrast, the agonist binding sites in homomeric nAChRs are formed by two adjacent alpha subunits.

Until recently, structural features of nicotinic receptor have been studied based on computational simulations of homologous proteins (77-80). However, a crystal structure of $(\alpha4)_2(\beta2)_3$ nicotinic receptor has been recently reported which details the ligand binding pockets along with the structure of the receptor (75). Two agonist binding sites have been located between alpha and beta subunits and a binding pocket is contributed by six loops of amino acids residues, namely loops A, B, C, D, E and F. Although, each subunit contains these six loops in its N-terminus, the positive side of the binding pockets is formed by the loops A-C of the alpha subunit and the negative side is contributed by the loops D-F of the beta subunit. Although loop F does not directly form the nicotine binding pocket, it might stabilize loop C though a hydrogen bond of D170 of loop F to the backbone nitrogen of C199 of loop C. The remaining loops (A-E) form a tight hydrophobic box to encapsulate a nicotine molecule where the different walls are formed as follows:

i. back wall: W156 (of loop B) and L121 (of loop E)

ii. front wall: C199, C200, Y197 and Y204 (loop C)

iii. floor wall: Y100 (loop A) and L121 (loop D)

iv. top wall: V111 and V119 (loop E)

Nicotine forms a cation- π interaction and a hydrogen bond inside the binding pocket. The hydrogen bond is formed between the pyrrolidine nitrogen of nicotine and backbone carbonyl oxygen of W156 of loop B, and the cation-π occurs between the same pyrrolidine nitrogen and the indole ring of W156. The conserved aromatic amino acid residues located in the beta-alpha and beta-beta interfaces undergo reorganization prohibiting the binding of nicotine in those interfaces. Nicotinic receptors are mainly cationic channels and flux Na^+, K^+, and Ca^{2+} ions. However, heteromeric receptors are less permeable to calcium ions than homomeric ones (57).

1.5.3 Nicotinic Receptor Subtypes

In the brain, twelve nicotinic receptor subunits (nine alpha subunits (α2-α10) and three beta subunits (β2-β4)) have been discovered (81,82). Since an alpha subunit, itself and in combination with one or more beta subunits, can form fully functional receptors, a vast variety of the assembled receptors is possible. In homomeric receptors, the principle and complementary side of a ligand binding pocket is composed of two adjacent identical subunits (83). In heteromeric receptors, the principle component can be made of α2 to α4 and α6 subunits, and the complementary component can be either a β2 or β4 subunit. The α5 and β3 subunits do not participate in the formation of a binding pocket, and hence, these

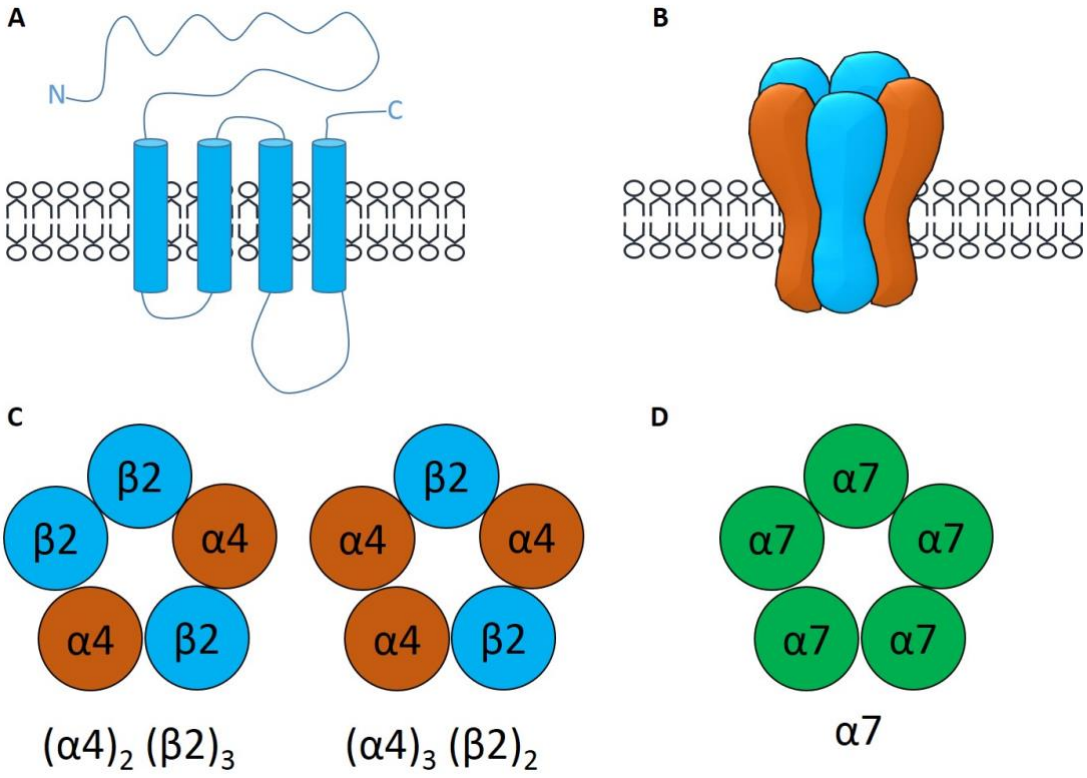

Figure 1.5 The structural features of nicotinic acetyl choline receptors (nAChRs). **(**A) Each subunit contains four transmembrane domains (M1-M4), a long extracellular N- terminal domain (~200 amino acids), a variable size intracellular domain between transmembrane domains M3 and M4, and a small C-terminal domain (4-28 amino acids). (B) A nAChR consists of five subunits which symmetrically arrange to generate a central ion pore that can transfer cations (Na^+, K^+, and Ca^{2+}). (C) An example of a heteromeric nicotinic receptor is an α4β2 receptor which can have one of the two different stoichiometric assembles: $(\alpha4)_2(\beta2)_3$ or $(\alpha4)_3(\beta2)_2$. (D) An example of homomeric nicotinic receptors is α7 whose all five subunits are identical.

subunits are called auxiliary subunits (84). The three most common subtypes of nicotinic receptors found in the brain are α4β2, α3β4 and α7.

1.5.3.1 α4β2 Nicotinic Acetylcholine Receptor*

α4β2* (* means one or multiple other nicotinic receptor subunits might present in the structure) nicotinic receptor subtypes are the most abundant nicotinic receptors in the brain and display the highest affinity to nicotine (85). These receptors contain two α4, two β2 and an α4 or β2 subunit in the fifth position leading to the possibility of two stoichiometries of the receptor. The stoichiometric isoform with three alpha and two beta subunits has an EC_{50} (half maximal effective concentration) of about 100 μM for acetylcholine and is termed as a low sensitivity isoform (75). The other possible isoform exhibited an EC_{50} value of 1 μM for acetylcholine and is called a high sensitivity isoform (86). It has been reported that the α4 subunit is associated with calcium permeability and a higher proportion of α4 subunits in a stoichiometry leads to higher calcium permeability (87). Hence, the low sensitivity isoform possessing three α4 subunits per receptor has higher calcium ion permeability and higher single channel conductance. Over all, α4β2 has lower calcium ion conductivity than α7 and α3β4 receptors (88). The α4β2 nicotinic receptor is the most abundant nicotinic receptor in the brain, has a very high affinity for nicotine and was reported to be upregulated with chronic treatment of nicotine (89,90). An alpha subunit makes the positive side of an agonist binding pocket while a beta subunit makes the negative side. No binding pocket was reported at the β2-α4 and β2-β2 interfaces. This finding was attributed to the fact that the beta subunits reorganize in the possible binding sites making impossible for ligands to bind (75).

α4β2 nicotinic receptors are expressed in different parts of a brain including the cortex, hippocampus, thalamus, hypothalamus, striatum, cerebellum, ventral tegmental area, substantia nigra, medial habenula, amygdala and interpeduncular nucleus (IPN) (80). These receptors have been associated with nicotine addiction which was validated employing α4 and β2 knock-out rodent models (91,92). β2 knock-out mice have been reported to self-administer cocaine solution but not nicotine solution when cocaine was replaced with nicotine, which indicated β2 subunit containing receptors are necessary for nicotine self-administration (91). Nicotine induces dopamine release in the brain (93), but when α4 knock-out mice was treated with nicotine, striatal dopamine level was found to be unchanged (92). This result indicated that α4 subunit containing receptors are necessary for nicotine induced dopamine release. Together, these studies implied that α4 and β2 containing nicotinic receptors might be involved in the nicotine dependency (94).

1.5.3.2 α7 Nicotinic Acetylcholine Receptor*

The α7* nicotinic receptor is the second most abundant nicotinic receptor in the brain and is generally considered to be a homo-pentameric receptors with 5 identical alpha subunits (95,96). This receptor has higher α-Bungarotoxin binding affinity compared to other nicotinic receptors and has 5 ligand binding pockets located in the interfaces between adjacent α7 subunits (97-99). The functional and structural properties of this receptor are usually conducted in *in vitro* cellular models with an α7 DNA only (77,95,100,101). In the brain, since other types of nicotinic receptor subunits are accessible, the structure of an α7 nicotinic receptor may contain one or more other types of nicotinic receptor subunits. Recent studies have demonstrated the presence of β2 subunits with α7 subunits in the brains

of rodent models (102,103). The ratio of these subunits in a functional receptor was determined *in vitro* by forming concatemers of α7 subunits with β2 subunits (104). The results demonstrated that an α7 receptor can have up to three β2 subunits in a functional receptor and with the increase in the number of β2 subunits, the duration of channel opening increases. These authors have hypothesized that with the increase in the number of β2 subunits, the desensitization decreases which leaded to the increased channel opening time (104). The α7 nicotinic receptor has been reported to possess fast activation, fast desensitization and high calcium permeability (76,88,105). This receptor has been reported to be expressed in cortex, hippocampus, hypothalamus, medial habenula, cerebellum, interpeduncular nucleus, ventral tegmental area and amygdala (80).

1.5.3.3 α3β4 Nicotinic Acetylcholine Receptor*

α3β4* nicotinic receptors are expressed in the central and peripheral nervous system. In the central nervous system, most of the α3β4* nicotinic receptors are expressed in the interpeduncular nucleus and medial habenula but a moderate to high expression was reported in thalamus, cerebellum and substantia nigra (106). α3β4 nicotinic receptors are hetero-pentameric receptors and can be assembled into two different stoichiometric isoforms. One isoform can have two α3 subunits and three β4 subunits while other isoform can have three α3 subunits and two β4 subunits. This receptor has displayed moderate upregulation with 10 μM nicotine treatment and a 5-fold upregulation with 1 mM nicotine treatment (107). It is relevant to mention that the physiological concentration of nicotine was reported to be 25-444 nM (108) and no upregulation of this receptor was reported with the presence of the physiological relevant concentrations of nicotine. Due to their possible

contribution in nicotine dependency, in recent years, more attention has been devoted to study the effect of nicotine into α3β4* nicotinic receptors (72,109-113). Studies with β4 knock-out animal displayed that this subunit is necessary for nicotine induced hypolocomotion, antinociception and hypothermic response (114,115). The absence of β4 subunits in mice led to reduced nicotine withdrawal somatic signs which are measured in mice with the presence of shaking of body or head, scratching, and grooming (106).

1.5.4 Upregulation of Nicotinic Receptors

Cellular function is regulated by a large number of factors including response to external stimuli often through membrane receptor activity. One way cells control activity is through the modulation of the numbers of membrane receptors on the cell surface. In the presence of high concentrations of agonist, some membrane receptors are often down regulated, and , in contrast, in the presence of antagonist, the target receptors are upregulated (116,117). An opposite result has been reported when α4β2 nicotinic receptors were chronically treated with a nicotinic receptor agonist such as nicotine (81,117). Upregulation of receptors is defined as an increase in the total number of receptors in the cell surface while down-regulation is the opposite of upregulation (81). An agonist binds and activates a receptor while an antagonist blocks the effect induced by an agonist. Both agonist and antagonist have been reported to upregulate nicotinic receptors in brains (118). The following subsections describe the effect of nicotine and other nicotinic receptor ligands in the upregulation of nicotinic receptors along with the possible mechanisms of upregulation.

1.5.4.1 Ligand-Induced Upregulation

Nicotine induced upregulation was discovered in 1988 by postmortem analysis of the brains of smokers in comparison with that of non-smokers (119). The result indicated a significant increase in the number of nicotinic receptors in the smoker's brain compared to the non-smoker's brain. (119). This result indicated that the nicotine causes upregulation of the receptors in the brain. When rodents were repeatedly treated with nicotine, the binding of the radio labeled agonists of α4β2 nicotinic receptors, nicotine and acetylcholine, were significantly increased indicating nicotine induced upregulation of the receptors (120,121). Nicotine induced upregulation is not uniform over the different regions of the brain. In rodent models, nicotine induced upregulation was reported in several brain regions including the cortex, midbrain, hindbrain, hypothalamus, and cerebellum (121) while some other regions displayed resistance to nicotine induced upregulation including the striatum (121) and thalamus (122). Nicotine induced upregulation has been reported to be dose dependent. In clonal cell culture, as low as 100 nM nicotine has been reported to cause upregulation and 10 μM nicotine caused a 15-fold increase in the α4β2 nicotinic receptor level (123). Nicotine induced upregulation is a transient event and the upregulated receptors usually relapse back to the basal level with varying time frame depending on the model being studied . For example, mice took 7-10 days to reach the basal level of nicotinic receptors upon cessation of nicotine treatment while rat and human have been reported to take 10-15 and 21-60 days to reach the basal levels of nicotinic receptors (81).

In addition to nicotine, some other nicotinic receptor ligands including (–)-Cytisine (here after Cytisine), Varenicline and Bupropion have been reported to alter the expression level of nicotinic receptors. Cytisine is a natural product extracted from the seed of the plant *Cytisus laborinum* (124). Cytisine acts as a partial agonist for α4β2 nicotinic receptors (125), and this ligand has been reported to upregulate as well as alter the stoichiometry of α4β2 nicotinic receptors. This ligand has also been used as a smoking cessation agent in Central and Eastern Europe for over 40 years (125). Varenicline is a synthetic analog of Cytisine and was developed and marketed as a smoking cessation agent by Pfizer (126). Varenicline is also a partial agonist for α4β2 and agonist for α7 nicotinic receptors (127). This ligand has been reported to upregulate α4β2 nicotinic receptors (128). Bupropion was originally approved by the FDA as an antidepressant agent but it has recently been approved by the FDA as a smoking cessation agent (129,130). Bupropion is an antagonist of α4β2 and α7 nicotinic receptor and displayed 12-times more efficacy for blocking α4β2 than α7 (115).

1.5.4.2 Upregulation Mechanisms

Understanding the mechanism of ligand induced upregulation is currently an area of active research, and a number of models have been proposed to explain the mechanism of upregulation. One such mechanism involves desensitization of nicotinic receptors (131,132). This model assumes that the desensitized state of a nicotinic receptor induced by repeated exposure to nicotine somehow causes the upregulation of the receptors (118). Recent reports have discounted this mechanism because nicotinic receptors have been reported to be upregulated at a very low concentrations of nicotine which activate and

desensitize an insignificant proportion of receptors. For example, α4β2 nicotinic receptors have been reported to be upregulated by 100 nM nicotine (123). Under these conditions, only 4 % of the high sensitivity isoform and 0 % of the low sensitivity isoform of α4β2 are activated and desensitized (118). However, when amino acids located in the binding pocket of nicotinic receptors were mutated to diminish ligand binding, nicotine induced upregulation disappeared indicating that ligand binding is necessary for upregulation of nicotinic receptors (133) regardless of the need for desensitization of the channel.

Another model describes the upregulation based on nicotine induced reduction in the turnover rate of nicotinic receptors (134). This model assumes that nicotine interacts with the cell surface nicotinic receptors to increase their stability, thus, reducing their internalization and degradation. An initial study with physiological relevant nicotine concentrations (500 nM) and cell surface receptor biotinylation reported a half-life for nicotinic receptors on the plasma membrane of 62.8 and 12.6 hours with and without the presence of nicotine, respectively (134). This study clearly validated this model of upregulation. However, subsequent studies were not in agreement with this study (135-137). Hence, further experimentation is necessary to deduce if reduced turnover rates are associated with nicotine induced upregulation.

One of the prominent models of nicotine induced upregulation of nicotinic receptors is known as “Inside-out Pharmacology” (118). According to this model, first, nicotine crosses the plasma membrane to enter intracellular organelles (ER and Golgi) (134,138). In these organelles, nicotine acts as a maturation enhancer or pharmaceutical matchmaker by helping the dimers and trimers of nicotinic receptor subunits to assemble

into pentamers (138). Second, the assembled receptor remains stabilized in the ER through interaction with the nicotine. Nicotine also helps the nicotinic receptors to be packed into COPII vesicles in the ER and thus assists the receptors to be transported to the Golgi (139). Third, in the Golgi, some upregulated receptors exit to reach to the plasma membrane while some other upregulated receptors are assumed to fail the quality control check and go back to ER through COPI vesicles (140). This retrograded transportation from the Golgi to the ER is necessary for upregulation. It is believed that nicotine stays bound with the nicotinic receptors when located in the COPI or COPII vesicles to keep the receptors desensitized (118). Thus, nicotine increases the density of stable nicotinic receptors in the secretory pathway leading to an increase in the delivery of COPII vesicles loaded with receptors from the ER to the plasma membrane (141). Finally, nicotine might stabilize the receptors present on the plasma membrane reducing the turnover rate of the receptor and increasing the total number of receptors on the plasma membrane (142).

Regardless of the mechanism of ligand induced upregulation, nicotine can increase the total number of receptors in the plasma membrane, can alter the assembly of the $\alpha4\beta2$ nicotinic receptors, can increase the rate of delivery of receptors through COPII vesicles to the plasma membrane. Therefore, it is necessary to understand the mechanism of nicotinic induced upregulation and the change in stoichiometric assembly to understand nicotine dependency.

1.6 Drug Delivery Vehicles

1.6.1 Concept of Drug Delivery Vehicles

When therapeutics are administered to a human body, those compounds typically reach to all parts of the body and interact with both the diseased and healthy tissue causing undesirable side effects. Researchers are trying to develop drug delivery vehicles which can remedy these issues associated with the side effects. An ideal drug delivery vehicle should be loadable with virtually any kind of drug molecule, should be able to avoid confiscation by the body's immune system, should reach the targeted site selectively, should stay and continuously deliver drugs in the targeted site for a prolonged period of time, and should be biocompatible and biodegradable. A number of approaches has been used to develop ideal drug delivery vehicles including lipid-based systems (liposomes, exosomes and cell-derived vesicles), polymer-based systems (pegylation, Chitosan, PLGA), nanoparticles formation, and absorption in carbon nanotubes or mesoporous silica (143-150). A brief description of the most common approaches is as follows:

1.6.2 Liposomes

Liposomes are defined as spherical nanovesicles consisting of at least one lipid bilayer encapsulating an aqueous medium in the center (Figure 1.6) (151,152). The bilayer can be made of either natural lipids or synthetic amphiphiles in conjugation with cholesterol which provide stability and membrane permeability (152-154). These vesicles are the most widely studied nano carrier for delivering therapeutics to a targeted location

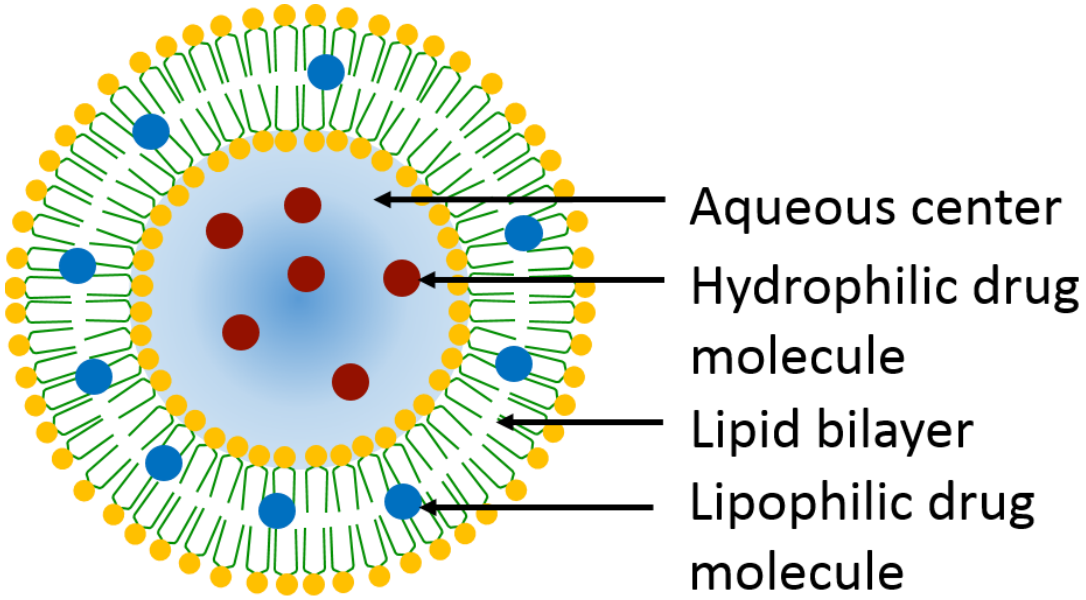

Figure 1.6 Loading of hydrophilic and lipophilic drug molecules into liposomes. Liposomes are spherical vesicles consisting of at least one lipid bilayer with an aqueous center. Hydrophilic drug molecules can be loaded into the aqueous layer and lipophilic drug molecules can be incorporated into the lipid bilayer.

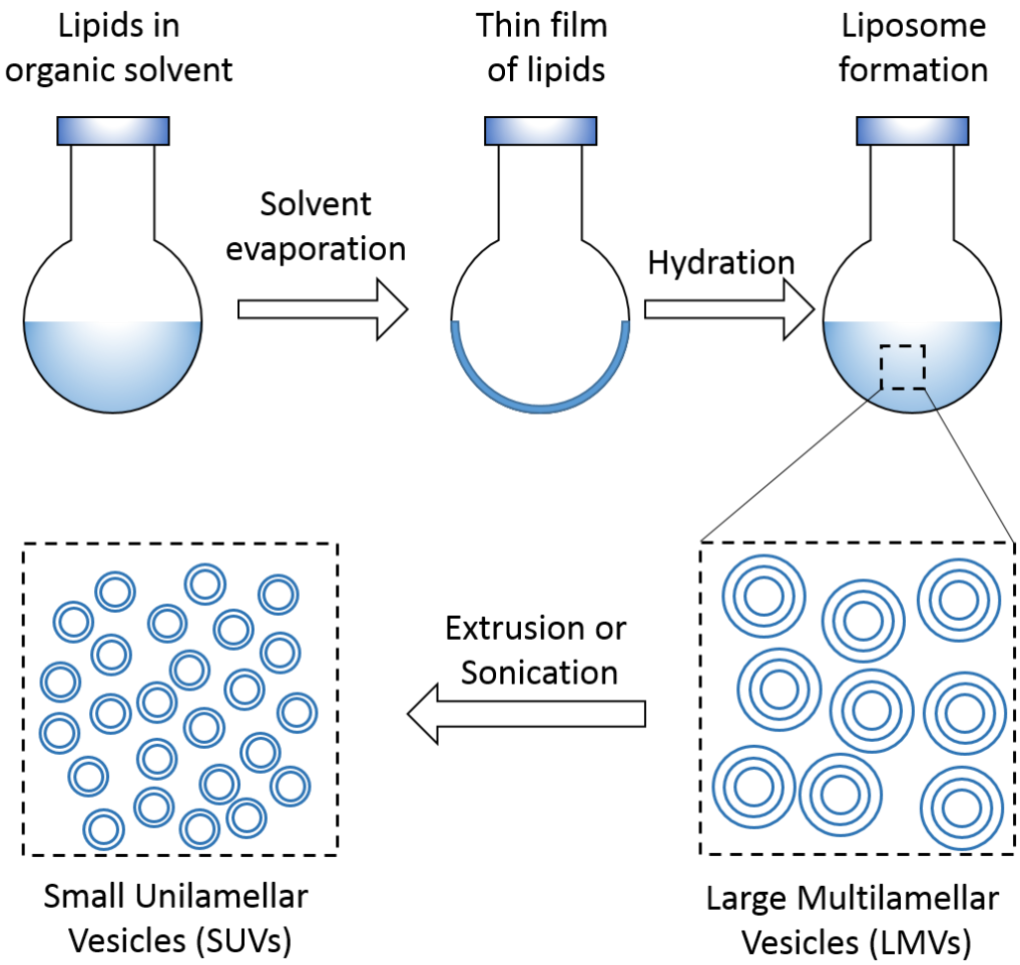

Figure 1.7 The thin film dehydration method of the preparation of liposomes. At first, lipids are dissolved in an organic solvent and the solvent is evaporated using rotary evaporation producing a thin film of lipids on the surface of the container. A buffer is added to the flask under vigorous shaking to rehydrate the lipid film generating large mulilamellar vesicles (LMVs). Extrusion or sonication is applied to reduce the size of the vesicles and these new vesicles are termed as small unilameller vesicles (SUVs). Both LMVs and SUVs are called liposomes.

and can encapsulate both lipophilic and hydrophilic therapeutic molecules making them ideal vehicles for a wide variety of molecules (151). The hydrophobic molecules are solubilized in the lipids of the bilayer while hydrophilic molecules are encapsulated in the aqueous phase located in the center of the liposomes (155,156). These vesicles can enhance the therapeutic indices of different drug molecules by altering their pharmacokinetic and pharmacodynamics properties (157). Liposomes can protect the encapsulated molecules from fast degradation, inactivation, and dilution during circulating in the body fluid (158). These vesicles can be produced with varying charge, size and lipid content. The charge of the vesicles is an important determinant of the function of the liposomes. For example, negatively charged liposomes repulse each other prohibiting them from aggregating while positively charged liposomes attract negatively charged lipids of the cell membrane providing better internalization (159,160). The most common method of liposome preparation is the Bangham method (Figure 1.7) (161). In this methods, the lipids with or without drug molecules are dissolved into an organic solvent and the resultant mixture is dried on the surface of a round-bottom flask by evaporating the solvent using a rotary evaporator. This process leaves a film on the surface of the flask and this film is rehydrated with vigorous shaking. The shaking process is considered as a critical determinant of the size of the liposomes; mild shaking generates larger vesicles while vigorous shaking produces smaller vesicles of non-uniform size (162,163). Sonication in a water bath and extrusion through polycarbonate membranes are the most common methods applied to reduce the size of liposomes. The size of these vesicles depends on the size of the membrane pore and the number of extrusion cycles through the membrane (164,165).

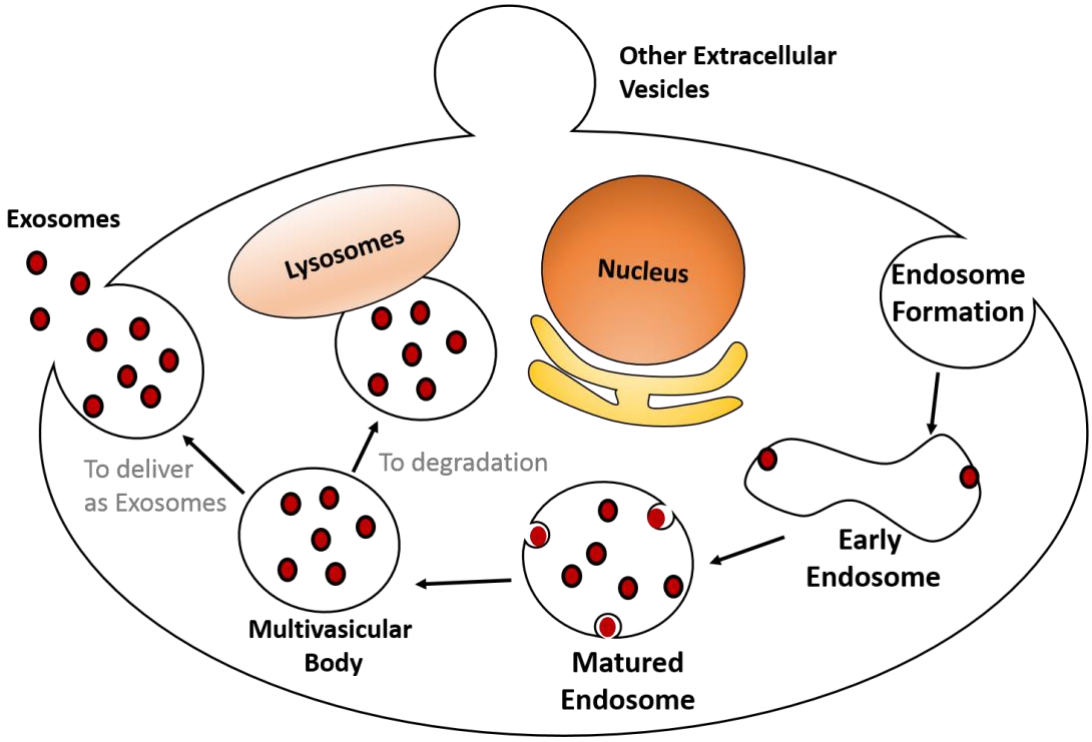

Figure 1.8 Mechanism of exosome formation in a cell. An endosome is formed by inward budding of the cell membrane. The early endosome is matured and intraluminal vesicles are generated by budding in the membrane of the endosome. Matured endosomes containing intraluminal vesicles are called multivesicular bodies which are either degraded by fusing with lysosomes or excretes exosomes by fusing with the plasma membrane. The other extracellular vesicles are produced by budding out from the plasma membrane.

Based on the choice of phospholipids, liposomes can be cationic, anionic or neutral and the drugs can be loaded into the liposome during or after the preparation of the vesicles.

1.6.3 Exosomes and Other Extracellular Vesicles

Exosomes are spherical vesicles that are excreted by a cell to the extracellular region and which encapsulate cytosol in their centers (1). These vesicles are considered to be a second means of communication between cells located at a distance while the first means of communication is the secretion of hormones. Exosomes deliver DNA, RNA, proteins and/or lipids from the donor to the acceptor cells. These vesicles are nano scale in size (30-100 nm) and are produced with the following mechanism (Figure 1.8) (1,166-168). At first, an endocytic vesicle is generated from the plasma membrane of the cell and the endocytic vesicles are matured which then produce inward budding to make small vesicles inside the lumen. The matured endosomes containing intraluminal vesicles are called multivesicular bodies (MVBs). The MVBs either fuse with the lysosomes for degradation or fuse into the plasma membrane of the cell releasing all the intraluminal vesicles into the extracellular environment (169,170). The intraluminal vesicles released outside of the body are termed as exosomes. Exosomes differ from the other types of extracellular vesicle – microvesicles, microparticles, ectosomes, oncosomes– according to the origin and size of the vesicles. While exosomes are produced through the generation of multivesicular bodies and have size of 30-100 nm in diameter, the other extracellular vesicles are generated by budding out from the cell membrane and are usually larger in size than exosomes (1) Since exosomes and other extracellular vesicles are naturally present in the body, they are being studied to employ as a drug delivery vehicle (171). The characteristic size and content of

exosomes are being extensively studied to utilize them for drug delivery or diagnosis purposes (3,172-174). These exosomes possess the potential of being loaded with drug molecules to selectively deliver them to a targeted location. However, the production of exosomes is a very slow and time consuming, and thus an expensive process.

1.6.4 Polymeric Nanoparticles

Drug molecules can be incorporated into polymers to produce nanoparticles. One such approach is the conjugation of therapeutics into linear or branched polyethylene glycol (PEG) polymers. The process of conjugating PEG into a molecule of interest is called pegylation; this process increases the size of the molecules and thus reduces the renal clearance as the kidney eliminates molecules by filtering according to their size. This formulation has been reported to possess low immunogenicity, toxicity and antigenicity (175,176). PEG polymers have displayed improved pharmacokinetics and pharmacodynamics of drugs (175-177). This polymer is removed from the body though renal excretion when the molecular weight of PEG is less than 30 kDa and via fecal when the size of the PEG is greater than 20 kDa.

Another approach of polymeric nanoparticle formation involves conjugation of therapeutics with a poly lactic-co-glycolic acid (PLGA) consisting of repeating units of lactic acid and glycolic acid. Hydrolysis of PLGA generates the monomers- lactic acid and glycolic acid- which being endogenous in the body easily become digested exhibiting minimal toxicity. The FDA has approved PLGA as a drug carrier because of its biocompatibility and biodegradability. The rate of PLGA elimination from the body

depends on both the molecular weight of the polymer and ratio of the two monomers (178). The most common methods to form PLGA nanoparticles loaded with drug molecules are single or double emulsion solvent evaporation techniques (179,180). In single emulsion techniques, PLGA is dissolved in an organic solvent and drug molecules are dissolved in the PLGA solution if the drug is hydrophobic. This organic solution (suspended phase) is added into water (continuous phase, higher in volume than the suspended phase) and sonication or another homogenization method is applied to make droplets of the organic phase (called oil phase, o) in water (w) in the presence of an emulsifier (a surface acting agent containing hydrophilic head and hydrophobic tail group). The emulsifier helps to stabilize the droplet in the water. The solvent of the emulsion is allowed to evaporate, leaving behind the oil-in-water (o/w) emulsion. Similarly, hydrophilic drug molecules can be encapsulated into a water-in-oil (w/o) emulsion by dissolving the drug into the water phase and making the water a suspended phase and the organic solvent a continuous phase. In double emulsion techniques, the emulsion generated in single emulsification techniques is emulsified into another solvent, water or oil, based on the outer layer of the previous emulsification. For example, water-in-oil can be emulsified into water or organic solvent to produce water-in-oil-water or water-in-oil-in-oil emulsions respectively. Water-in-oil-in-water (181,182) and solid-in-oil-in-oil (183-185) double emulsion techniques have been reported to encapsulate therapeutic peptides into nano- or micro-particles for targeted delivery or extended release of the cargo.

The most frequently employed polymer to encapsulate therapeutic peptides or to coat other types of drug delivery vehicles is chitosan obtained from a naturally occurring polysaccharide, chitin, by deacetylation. This polymer is biodegradable, biocompatible and

nontoxic making them suitable as drug delivery vehicles (186-189). Chitosan nanoparticles are insoluble in aqueous solution of neutral pH but soluble in acidic solutions of pH less than 5. At low pH, a chitosan polymer become positively changed due to the protonation of an amino group, and this protonated chitosan can bind with the mucosa of the epithelial cells of lungs allowing extended release of the incorporated drug molecules (190-192). Therapeutics are usually loaded into chitosan particles during preparation of the particles. There are a number of methods available to prepare chitosan nanoparticles loaded with drug molecules including ionic cross-linking, covalent cross-linking, precipitation, polymerization, self-assembly, and spray drying (189). The entrapped therapeutics become available in the body when the particles become depolymerized in the presence of lysozyme. This characteristic degradation of the particles depends on the molecular weight, degree of deacetylation, hydrolysis time and availability of amino groups of the chitosan (193). A higher degree of deacetylation leads to faster degradation (194).

However, conjugation of therapeutics with polymer has been reported to elicit immune response which leads to a rapid clearance of the pharmaceuticals (195). Additionally, this approach can cause therapeutics to bind with serum proteins and to reduce uptake by the target tissue.

1.6.5 Absorption in Nanoparticles

Mesoporous silica nanoparticles can also be used as a drug delivery system. These particles contain honeycomb-like pores providing high surface area and pore area. Drug molecules are absorbed inside the pores of the mesoporous particles. A number of

modifications have been introduced to alter their surface properties including coating with positively charged amines (196), cell permeable chitosan polymers (181), and pH sensitive polymers (197,198) to load and deliver therapeutics.

1.6.6 Advantages and Applicability of Cell-Derived Vesicles

Some researchers used the term cell-derived vesicles as a synonym for extracellular vesicles (199-201). For our discussion in this work, I define cell-derived vesicles as spherical vesicles generated form the ER and plasma membrane through homogenization or extrusion of cells. Since the human body has a natural tolerance for exosomes which are present in the body and used by cells to communicate with non-adjacent cells, some researchers are utilizing cell derived vesicles, which are structurally similar to exosomes, to deliver therapeutics into targeted locations (39,40). These studies produce cancer cell-derived vesicles by extrusion or spinning cups which were labeled with lipophilic dye molecules. These dye labeled vesicles were then injected into the tail vein of a mouse model containing a tumor, and the injected vesicles reached the tumor. This approach validates the concept that cell derived vesicles can be employed to deliver chemotherapeutics selectively to a tumor. However, the traditional extrusion or spin cup approach cannot simultaneously prepare and load vesicles with drug molecules. Thus, a secondary approach, (for example, remote loading) is necessary to accomplish the loading (202). We prepare cell-derived vesicles with nitrogen cavitation where the cells are fragmented and the cell fragments spontaneously reorganize to produce vesicles encapsulating the cell suspension solution. Therefore, by maintaining the drug molecules in the suspension

solution, we can simultaneously generate and load vesicles which is more efficient, faster and cheaper process.

Chapter 2

Cell Derived Vesicles for Single Molecule Imaging of Membrane Proteins

2.1 Introduction

Complex protein structures, such as membrane receptors, regulate many aspects of cellular function including the initiation of signal transduction pathways (203,204). Transient interactions between proteins often complicate efforts to fully understand the function of specific biomolecules (205). The primary challenge is that a single species can exist in multiple conformational or functional states. Single molecule approaches are often employed to resolve these dynamics as they avoid ensemble averaging across multiple states (206), but the physiological concentration of receptors is often too high for single molecule measurements. Additionally, isolating individual proteins poses a challenge for receptors as they tend to aggregate on the cell surface. A common approach to overcome this concentration barrier (207,208) is to isolate biomolecules by purification from the cell (28,209). This approach only works for proteins that can either be solubilized or stabilized in a detergent solution. As a result, a variety of approaches have been employed to isolate receptors or to apply single molecule techniques in cells (10,207,208,210). For example, sub-micron liposomal vesicles composed of artificial bilayers have been utilized as nanocontainers for the isolation of proteins (18). This approach requires receptors to be temporarily supported in a detergent solution. This induces a major disadvantage in the receptors which are completely removed from their physiological environment.

Here we introduce a single molecule approach that isolates receptors in vesicles generated from cell membranes. Microsomes and other cell derived vesicles are widely used for biochemical applications to study membrane receptors and other proteins (211,212). We utilize a similar strategy by generating vesicles from cells expressing a

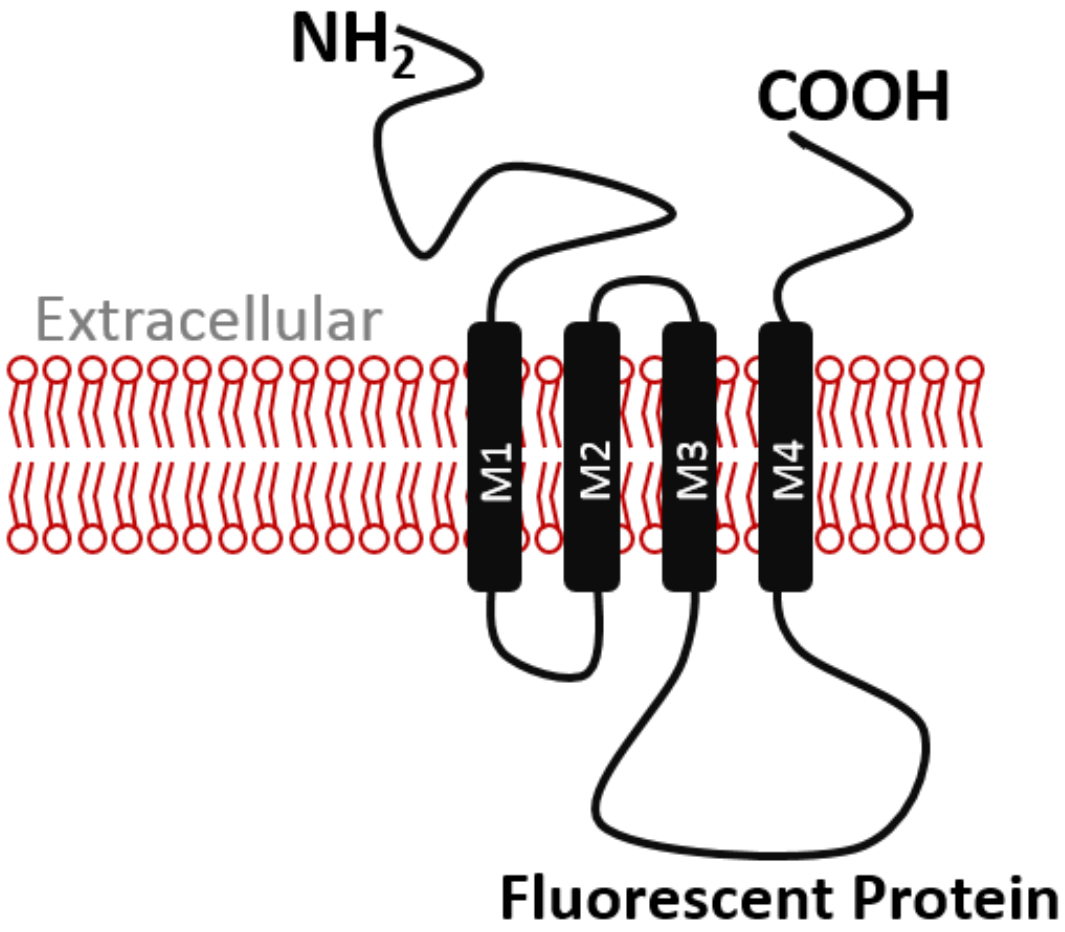

Figure 2.1 A simplified representation of the different domains of a nicotinic receptor subunit. A fluorescent protein (e.g. GFP, mCherry, etc.) has been incorporated into the intracellular region between transmembrane domain 3 (M3) and 4 (M4). A super ecliptic pHluorin (SEP) was incorporated at the C-terminus of the nicotinic receptor subunit.

protein of interest in order to perform single molecule imaging. This approach leaves membrane proteins inserted in the same physiological membrane in which they resided within the cell. The use of these vesicles eliminates the need to support proteins in a detergent environment or to encapsulate them in vesicles composed of artificial membranes. We demonstrate the versatility of this method using several classes of membrane receptors, and expand this technique to determine the stoichiometry of α3β4 nicotinic receptors.

2.2 Experimental Procedures

2.2.1 DNA Construct Preparations

In the nicotinic receptor subunits, fluorescent proteins were incorporated in the intracellular region between 3rd and 4th transmembrane domains (Figure 2.1). Epitope tags (Flag, HA, etc.) were added in the C-terminus of the amino acid sequence. EGFP (hereafter, GFP) was added to the C-terminus of CFTR and EGFR proteins separately

2.2.2 Cell Culture

The human embryonic kidney 293T (HEK293T) cell line was obtained from Prof. Louis B Hersh, Department of Biochemistry, University of Kentucky and was maintained in a matrigel (Invitrogen) coated T75 flask with Dulbecco's Modified Eagle Medium (DMEM) supplemented with 1 % penicillin streptomycin (Invitrogen) and 10 % fetal bovine serum (Invitrogen). Approximately 3 million HEK293T cells were plated in a matrigel coated T75 flask 24 hours before transfection. Cells were transfected at 60-70 % confluency with lipofectamine 2000 (Invitrogen) according to the manufacturer's protocol

with slight modification. 14 µl lipofectamine 2000 was added to transfect a flask of cells. In order to lower the expression level of proteins, to facilitate single receptor isolation, we have lowered the amount of DNA plasmid of all alpha, beta and CFTR to 3.5 µg. During co-transfection of two different types of DNA plasmids, 3.5 µg of each DNA plasmid was used, except Seq61-mCherry and plasma membrane-mCherry (PM-mCherry). The plasma membrane and ER markers (i.e. PM-mCherry and Seq61-mCherry) were cotransfected with 500 ng of each plasmid. EGFR-GFP was expressed with 2 µg DNA plasmid per flask. Transfected cells were allowed to grow by incubating at 37 °C with 5 % CO_2 flow for 22-24 hours. Then, these cells were used to prepare vesicles.

2.2.3 Vesicle Preparation

At first, cells were visualized under a microscope to ensure that they were alive and healthy, and the desired fluorescent protein conjugated receptors had been expressed. Then, the transfection media was removed from the flask, and 5 ml Versene (Invitrogen) was added and incubated at 37 °C for 5 minutes. Versene is an EDTA based chelating agent which is used to dissociate cells from flasks. Afterward, the cell slurry was collected in a 15 ml tube and centrifuged at 400 ×g for 5 minutes. The obtained pellet was resuspended in 5ml Sucrose-Protease inhibitor buffer (10mM HEPES pH 7.5, 250mM Sucrose, protease inhibitor cocktail tablet (1 tablet per 10 ml buffer)) and placed into a precooled nitrogen cavitation chamber (Figure 2.2) (Parr Instruments Company, IL, USA). Nitrogen gas was flowed to the chamber for 250 psi, and the pressure was kept constant for 5 minutes. Then,

Figure 2.2 A nitrogen cavitation chamber employed to generate vesicles from mammalian cells. Cells were placed inside the chamber, the lid was closed and nitrogen gas was flowed into the chamber. The valve was opened to release the gas and to collect the cell lysate which contains spherical vesicles. This vesicles solution was purified with differential centrifugation.

the pressure was suddenly released, and the resulting cell lysate was collected in a 15 ml tube that was subjected to centrifugation at 4000×g, at 4 ℃ for 20 minutes. The pellet was discarded, and the supernatant was centrifuged at 10,000 rpm (9,800 × g), at 4 ℃ for 20 minutes. Once again, the pellet was discarded; the supernatant was centrifuged at 30,000 rpm (100,000 × g), at 4 ℃ for 1 or 2 hour(s). The pellet was rinsed with 400 µl of sucrose buffer (10 mM HEPES, 250 mM Sucrose pH 7.5) and resuspended in 400 to 800 µl of the same buffer. The solution containing vesicles was aliquoted and stored at -80 ℃.

2.2.4 Glass Bottom Dishes Cleaning

Glass bottom dishes (GBDs) were placed into 200 ml freshly prepared 5 M NaOH solution. This was incubated at room temperature for 1 hour or sonicated at 45 ℃ for 30 minutes. Then, NaOH was removed by washing with very high-velocity-flowing tap water for 30 seconds. About 1 ml ethanol (denatured) was placed into the dish, and was allowed to sit for 30 seconds and rinsed with tap water in the same way. This ethanol incubation and water-rinsing cycle was carried out for a total of three times. Then, GBDs were rinsed with DI water, and ethanol was sprayed on the top and bottom of coverslips of GBDs. Afterward, GBDs were dried with compressed air and subjected to the oxygen plasma clean for 5 minutes at the maximum level. The presence of background fluorescence in the GBD was checked under a microscope; if no background was detected, then the GBD was utilized to develop the immobilization system on it.

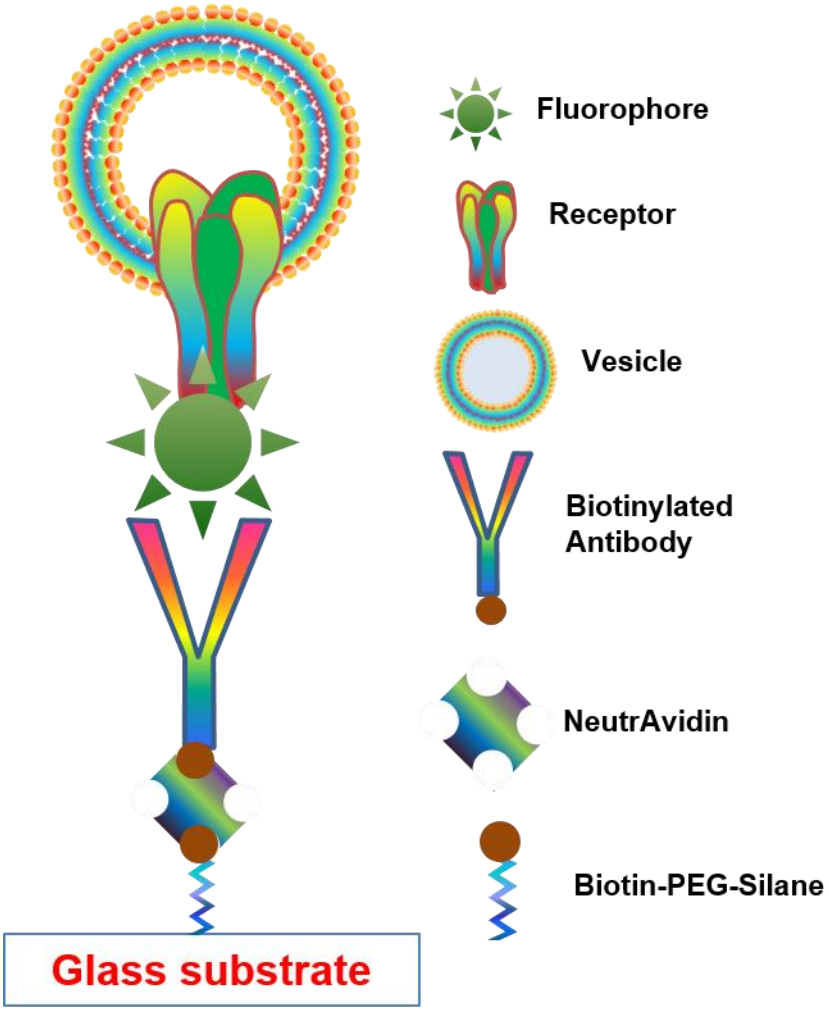

Figure 2.3 A schematic representation of different layers on top of a glass substrate to immobilize receptor inserted in vesicles. On the top of a glass substrate, first, a Silane-PEG-Biotin layer was placed, then NeutrAvidin, followed by biotinylated antibody and finally vesicles were immobilized.

2.2.5 Preparation of Immobilization System

Approximately 200 µl of 1 mg/ml of Biotin-PEG- Silane (MW 3500, Lysane Bio, Inc.) in a solution of 95% ethanol and 5 % water was added on top of the coverslip of a clean GBD. Then, it was incubated at room temperature for 30 minutes (Figure 2.3). Afterward, the unreacted reagent was removed by rinsing with denatured ethanol and DI water separately. Followed by, 200 µl of 0.10 mg/ml NeutrAvidin (Thermo Scientific) solution in PBS buffer was added on top of the GBD; it was incubated for 5 minutes at room temperature before rinsing (three times) with PBS buffer. Next, ~200 µl appropriate biotinylated antibody (1 µg/ml) was added on top of the GBD and incubated at room temperature for 15 minutes. The unbound antibody was removed by rinsing (three times) with PBS buffer, and about 1 ml PBS buffer was added into the GBD to keep the antibody in the solution. The antibody coated GBDs were examined under a microscope to verify no background fluorescence was present. GBDs with non-significant background fluorescent spots were used to immobilize vesicles.

2.2.6 Vesicle Immobilization

The freshly prepared Silane-PEG-Biotin-NeutrAvidin-biotinylated antibody coated glass bottom dishes (GBDs) (hereafter, termed as antibody coated GBDs) were employed to immobilize vesicles. About 100 times diluted vesicles were placed on top of the appropriate antibody coated GBD; it was incubated for 30 minutes at room temperature (Figure 2.3). Then, unbound vesicles were removed by rinsing (three times) with PBS buffer. These immobilized vesicles were imaged under ~1 ml PBS buffer.

2.2.7 Single Molecule TIRF Microscopy

488 nm and 561 nm laser lines were used to excite green fluorescent proteins (GFPs) and mCherry proteins respectively. The laser beam was directed through an oil-immersion type 1.49 NA Olympus ApoN 60X or 100X objective (Olympus America, Center Valley, PA, USA) mounted on an Olympus IX 81- inverted microscope (Olympus America) through corresponding excitation filter and dichroic. Intensities of the 488nm and 561nm lasers were adjusted to be ~20 W/cm^2 and ~40 W/cm^2 respectively. Low auto-fluorescence emitting immersion oil (Thorlabs, Inc.) was placed on top of the objective. A sample immobilized on a glass bottom dish was placed on top of the oil. An auto focus module (Model: IX2-ZDC2, Olympus America) was used in order to limit focal drift, and a stage control module (Olympus America,) was employed to remember the position of the field of view. The objective used is capable of TIRF microscopy; TIRF was achieved by translating the beam laterally across the objective lens. This was done by adjusting the angle of the excitation beam using a stepper motor. The emitted fluorescence was collected through the objective and dichroic, and directed to an electron multiplying CCD (Andor) camera through an appropriate emission filter. For each of the samples, 10 to 15 movies with 500 to 1000 frames (200 ms per frame) were taken. For dual color experiments of α3-mCherry β4-GFP, at first, movies were taken with the 561 nm laser and then with the 488 nm laser on the same field of view using the stage control module.

2.2.8 Photobleaching Step Analysis

The tiff files of single frames or multiple frames (also termed as movies) were analyzed with an open source software, ImageJ (NIH, USA). The background of an image

was subtracted using rolling ball background subtraction with 20 pixels radius. Circular region of interests (ROIs) with 4 pixels radius were selected manually, and Time Trace Analysis V2.0 (or V3.0) plugin was used to generate time traces. During the dual color experiments with GFP and mCherry conjugated proteins, ROIs were first selected on the movies corresponding to the mCherry channel, and then, translated to the movies corresponding to the GFP channels. A Matlab (Mathwork) script was written to automatically plot graphs corresponding to individual punctate on the movie, and the number of photobleaching steps were then determined manually (13,29). Time traces showing clear photobleaching steps were counted and those showing indistinct bleaching steps or exponential decay were discarded. Heteromeric nicotinic acetylcholine receptors form a pentameric structure composed of either 2 or 3 of each subunit. Identification of more than 3 bleaching steps was used as an indication of multiple receptors within a single vesicle or within the diffraction limit. For dual color experiments (of α3-mCherry β4-GFP), stoichiometry was assigned only to vesicles that showed a total of 5 bleaching steps (2-mCherry & 3-GFP) or (3-mCherry & 2-GFP).

2.2.9 Study the Ligand-Receptor Interaction via FCS

Epidermal growth factor receptor conjugated with GFP (EGFR-GFP) containing vesicles were generated by expressing HEK293T cells. Vesicles were diluted 20 times in the sucrose buffer from original preparation prior to the experiment (see section 2.2.3). . About100 µl of the sample was placed on the top of water immersion objective (LUMPlanFL N, 60X, Olympus), and the confocal laser beam was focused 50 microns into the sample solution. The laser power was held constant at 10 microwatts. The emitted light

was directed through the appropriate filter to a 50 μm fiber acting as a pin hole. The light was then directed to an APD (Avalanche Photodiode) τ-SPAD single photon counting module (PicoQuant GmBh, Berlin, Germany). A PicoHarp-300 time correlated single photon counting module was used to record the photon arrival time. Fluorescence correlation was determined using Symphotime 64 Software (PicoQuant GmBh). FCS data were recorded with (i) vesicles only, (ii) 0.3 nM epidermal growth factor conjugated with tetramethylrhodamine (EGF-TMR) (Life technology) only, and (iii) EGFR-GFP containing vesicles mixed with 0.3 nM EGF-TMR. 488 nm and 561 nm laser sources were employed to determine the diffusion times with GFP and TMR respectively.

2.2.10 Ligand-Receptor Interaction on the Surface

HEK293T cells were cotransfected with EGFR- GFP and CFTR-HA plasmids and vesicles were generated from them. 100 μl vesicles from our preparation was mixed with EGF-TMR (EGF conjugated with tetramethylrhodamine) so that the final concentration of the EGF was 1 μM. This mixture was subjected to dialysis to remove unbound EGF-TMR with a Slide-A-Layer MINI Dialysis Devices, 10K molecular weight cut-off (Thermo Scientific) for 4 hours at room temperature under stirring in 200 ml PBS buffer and buffer was changed in every 30 minutes. The purified mixture was diluted 100 times and added to a Biotin-NeutrAvidin- Anti HA antibody coated coverslip. A 488 nm laser in TIRF with the corresponding excitation and emission filters was used to visualize vesicles (by locating GFP) and a 561 nm laser with corresponding setup was used to locate the positions of TMR. Overlap of the images of the two channels indicates the binding of ligand (i.e. EFG) with the receptor (EGFR).

2.2.11 Determination of the Percentage of Vesicles Inserted with Receptors

Since the expression of receptors was kept very low to facilitate single receptors per vesicles, some of the vesicles might not contain a receptor. Determining proportion of vesicles containing receptors in them can be used to determine probability of isolating single receptor per vesicle. To do so, we have expressed HEK293T cells with (i) plasma membrane marker –plasma membrane mCherry (PM-mCherry), (ii) ER marker– Seq61 mCherry, and (ii) α3-GFP, β4-wt nicotinic receptor. Membrane markers were employed to locate vesicles, and receptors were incorporated to detect the positions of the receptors in vesicles. We have generated vesicles from the HEK293T cells, and then those vesicles were immobilized on an anti-mCherry antibody coated coverslip. Images were taken exciting mCherry (561 nm) and GFP (488 nm) separately on the same field of view. The number of particles were determined in both images to find the number of vesicles and receptors present in them.

2.2.12 Determination of the Size of Vesicles

To determine the size, α3-GFP β4-GFP containing vesicles were generated. An ISS Alba confocal fluctuation system coupled with a Nikon Ti-U inverse microscope with a water objective (60x, 1.2 NA) was used to find the size of the vesicle through FCS. A 488 nm laser was employed to illuminate the sample. The emitted light was directed to two photomultiplier tube (PMT) detectors through a 514 long pass filter. The focal volume of the objective was calibrated with a solution of a known concentration of Rhodamine 110 (Diffusion coefficient = 440 $\mu m^2 s^{-1}$). The laser was focused at 100 μm into the solution. FCS data were recorded for 1 minute and 10 measurements were performed to determine

the mean and standard deviation of the measurements. FCS data was analyzed with VistaVision 4.0.120 (ISS Alba). The Stokes-Einstein equation was employed to determine the diameter of vesicles assuming the viscosity of the aqueous sample was the same as that of water.

2.2.13 Determination of the Functional Assembly of Receptors in Vesicles

Functional assembly of receptors was examined though characteristic calcium flux of α7 nicotinic receptor. The protocol to prepare vesicles in order to carry out calcium flux experiment was adapted from Smith *et. al.* (213) with required modification. Briefly, vesicles were generated by expressing α7-GFP and β2-Flag into HEK293T cells. 200 µl vesicles was mixed with Fluo-8 AM (Assay Biotechnology Company, Inc., CA, USA) with a final concentration of 10 pM and then incubated at 35 °C for 30 minutes to facilitate the loading of Fluo-8 AM dyes inside the vesicles. Then, buffer-A (5 mM HEPES pH 7.4, 144 mM NaCl, 5 mM KCl, 1.2 mM $MgSO_4$ and 5 mM glucose) was employed to dilute the vesicles to 10 ml. Free Fluo-8 AM dye molecules were removed by centrifuging the diluted vesicles (22,000 × g for 1 hour). The pellet was rinsed with 1ml buffer-A and resuspended to 400 µl Buffer-A. 200 µl of this solution was added on top of an anti-Flag antibody coated glass bottom dish to immobilize the vesicles. An image was taken exciting GFP (488nm). Since activated Fluo-8 and GFPs have the same excitation and emission spectra, a continuous excitation was made, before activating Fluo-8, to bleach all GFPs. Then, 1ml solution containing 2mM $CaCl_2$ and 200 µM Acetylcholine (ACh) chloride (Sigma-Aldrich) was added on top of the immobilized vesicles maintaining the same field of view. An image was taken when exciting Fluo-8 at 488nm.

2.2.14 Statistical Analysis of the bleaching steps

The probability of observing photobleaching steps from the time traces were calculated fitting the data based on a binomial distribution(214). The probability of observing a specific number of photobleaching steps, for a receptor with fixed number of subunits is calculated using:

$$\mathbf{F(k,n,p) = \frac{n!}{k!(n-k)!}\, p^k (1-p)^{n-k}} \qquad \textbf{Eq. 2.1}$$

Where, n is the total number of subunits, k is the number of observed units, p is the probability of GFP being in an observable state, and F is the probability of observing k number of photobleaching steps from n number of subunits. The reported fraction of GFP that is in a visible state varies widely (0.64 to 0.9) across publications (215-217). Recent publications have shown that at least 90% of GFP matures fully and remains in a fluorescent state corresponding to p = 0.9 (216). nAChRs can potentially form multiple stoichiometries. Thus, α3β4 can form a pentamer with either two or three α3 subunits. This leads to a combination of $(\alpha3)_2(\beta4)_3$ and $(\alpha3)_3(\beta4)_2$ stoichiometries. Modeling the probability of observing a specific number (m) of photobleaching steps for the mixed stoichiometry case requires a combination of two binomial distributions for k=1, 2, and 3 for both F_1 and F_2. F_1 corresponds to the case when n_1 GFP labeled subunits are in a receptor and F_2 when there are n_2.

$$\mathbf{F_{tot} = a_1 \cdot F_1 + a_2 \cdot F_2} \qquad \textbf{Eq. 2.2}$$

Stoichiometry can exist in equal distributions or be biased toward one stoichiometry. The fraction of the stoichiometry, a_i, is used to weight the distribution based on a mixed stoichiometry (n_i). For example, the probability of observing 2 bleaching steps (i.e. k=2) of α3 subunits from $(\alpha3)_2$ $(\beta4)_3$ (i.e. n = 2) and $(\alpha3)_3$ $(\beta4)_2$ (i.e. n = 3) can be calculated with:

$$\mathbf{F = a_1 \cdot F(2, 2, p) + a_2 \cdot F(2, 3, p)} \qquad \textbf{Eq. 2.3}$$

A custom Matlab script was used to fit our data to a binomial distribution with mixed stoichiometry using p=0.90. The data fit well with a distribution of 75 % $(\alpha3)_2$ $(\beta4)_3$ and 25 % $(\alpha3)_3$ $(\beta4)_2$ subunits. The error bars for subunit distribution are based on counting events and are calculated as the square root of the counts (218).

2.3 Results and Discussion

2.3.1 Vesicle Characterization

Cell-derived vesicles can be utilized to isolate single receptors to conduct single molecule experiments using total internal reflection fluorescence (TIRF) microscopy and fluorescence correlation spectroscopy (FCS). It is necessary to characterize cell-derived vesicles before using them for single molecule studies. Hence, we characterized the size of the vesicles, specificity of the immobilization of the vesicles, the probability of having single receptors on the vesicles, and the applicability of the vesicles to study protein oligomerization.

First, to determine size, we made vesicles from HEK293T cells transfected with α3-GFP β4-GFP nicotinic receptors. Both subunits were labeled with GFP to maximize the fluorescence signal. Then, these vesicles were used to determine the diffusion times in solution by means of FCS. The mean diffusion time was employed to determine the diffusion coefficient and radius through the Stokes– Einstein equation. Ten measurements were made to get an average and standard deviation of the vesicle size. The diameter of the vesicles was found to be 180 ± 20 nm, where error indicates the standard deviation (Figure 2.4).

In order to determine the probability obtaining a single receptor in a vesicle, we can determine the percentage of vesicles that contains receptors. This was done by preparing vesicles with α3-GFP and β4-wt along with membrane markers (sec61-mCherry as an ER and plasma membrane-mCherry as a plasma membrane marker). The vesicles were immobilized on the surface of a glass substrate though anti-mCherry antibodies. Then, the number of vesicles were determined by taking an image by exciting mCherry, which represented the positions of vesicles (Figure 2.5A). Followed by, another image was captured on the same field of view by exciting GFPs which indicated locations of the receptors (Figure 2.5B). The overlay of the positions of vesicles and receptors indicates that some of the receptors presents into the vesicles. Afterward, the number of particles present on the images were counted using ImageJ (NIH, USA) and it was found that about 15 % of the vesicles contained receptors in them. Therefore, it can be deduced that there was 15 % probability of obtaining a receptor in a vesicle, and thus only 2 % vesicles will contain two receptors. This indicated that there is about 98 % probability of having a single receptor in a vesicle.

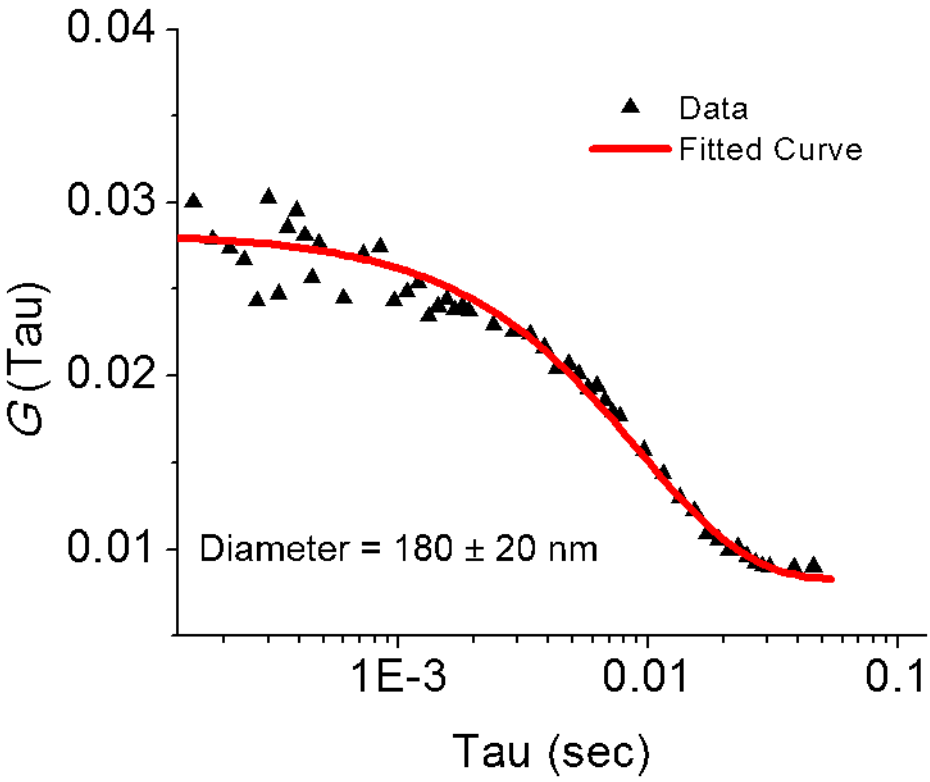

Figure 2.4 Determination of the size of vesicles using FCS. Vesicles inserted with α3-GFP β4-GFP nicotinic receptors were allowed to pass through the confocal beam of an FCS, and an auto correlation curve of the signal was obtained. The diffusion time extracted from the autocorrelation curve corresponded to a diameter of 180 nm.

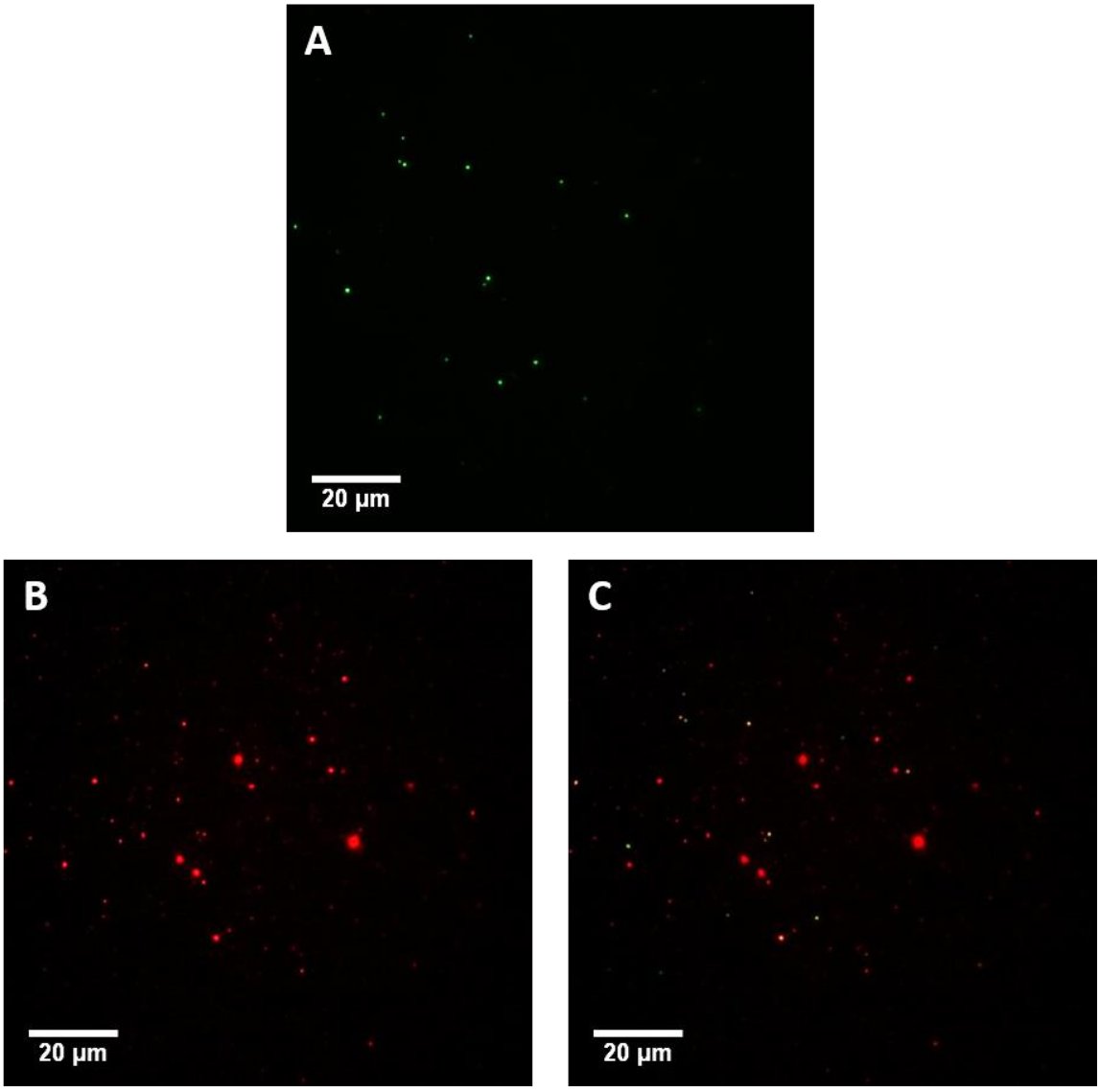

Figure 2.5 Determination of the number of vesicles inserted with receptors. Vesicles were generated with membrane markers conjugated with mCherry (to locate and count the number of vesicles) along with α3-GFP β4-wt nicotinic receptors (to find and count the number of receptors in the vesicles). (A) An image taken by exciting mCherry which indicating the positions of vesicles and (B) the position of receptors on the same field of view as this image was captured by exciting GFPs. The overlay (C) indicates some of the vesicles contain receptors in them. Counting the number of particles in image A and B, we found that ~15 % of vesicles contained receptors in them.

We also carried out a series of experiments to demonstrate that our observation was limited to the specific binding of receptors with the corresponding antibodies (Figure 2.6). We have prepared vesicles by transfecting CFTR-GFP and by co-transfecting CFTR-HA with EGFR-GFP. The surface tethered anti-GFP and anti-HA separately bound with CFTR-GFP and CFTR-HA respectively and this binding was verified by imaging the GFP tag with a TIRF microscope. However, HA tags did not attach with the anti-GFP antibody coated coverslip which indicated that vesicles were bound specifically on the surface.

We validated that we could isolate single receptors into the vesicles and the immobilization of vesicles are specific to the corresponding antibody. Next, to validate that cell-derived vesicles can be utilized to studying stoichiometric assembly of membrane proteins, we prepared vesicles with a membrane protein – CFTR– coupled with GFP (CFTR-GFP). A glass substrate was tethered with the anti-GFP antibody, and CFTR-GFP containing vesicles were immobilized on top of it (Figure 2.7). Movies were taken by exciting GFPs with 488 nm laser souce under a TIRF microscope. The obtained movies were analyzed to collect time traces from each peaks corresponding to single molecules. When time traces were plotted, it displayed mainly single or double steps photobleaching events. After analysis of photobleaching steps from about 10 movies, the data indicates the presence of 80 % single CFTR. Although, CFTR is primarily considered as a monomeric ion channel (219), it has been reported to present as a dimer as well (220). The presence of about 20 % two photobleaching steps might arise from the dimers of CFTR.

2.3.2 Analysis the Functional Assembly of Receptors in Vesicles

We have demonstrated that cell-derived vesicles have a size of about 200 nm in diameter, can isolate single receptors, can be selectively immobilized on a glass substrate and can be utilized to study stoichiometric assembly of membrane proteins. However, to validate that the membrane proteins isolated on the vesicles are functional, we have conducted the characteristic calcium flux experiment of nicotinic receptors. Vesicles were generated expressing α7-GFP β2-Flag nicotinic receptors in HEK293T cells and then loaded with Fluo-8 AM dye. The acetoxymethyl (AM) group present in the Fluo-8 dye helps it to penetrate through the cell membranes and enter inside the vesicles. As those vesicles were generated from a native cell membrane, we assumed that esterase enzymes would present in the vesicles, and this enzyme would remove the AM group of Fluo-8. The resulted Fluo-8 become charged allowing to bind with calcium ion and to produce fluorescence. The Fluo-8 AM loaded vesicles were immobilized on top of a coverslip with the help of an anti-Flag antibody. An image was taken by exciting GFPs to determine the positions of receptors in the field of view (Figure 2.8). Since GFP and activated Fluo-8 have similar excitation and emission spectra, a continuous excitation was applied to bleach all GFP molecules before activating Fluo-8 dyes. Then, acetylcholine and calcium ions are added on top of immobilized vesicles. Acetylcholine binds with the nicotinic receptor and activates the channel, and calcium ions can move through the open channel. Once calcium ions are inside the vesicles, they bind with Fluo-8 dye and activate the dye. Then, another image was captured exciting Fluo-8 (488 nm) on the same field of view as GFPs. The overlay image verified that all Fluo-8 spots were observed from the vesicles. Then the number of particles present at both images was calculated to determine the number of

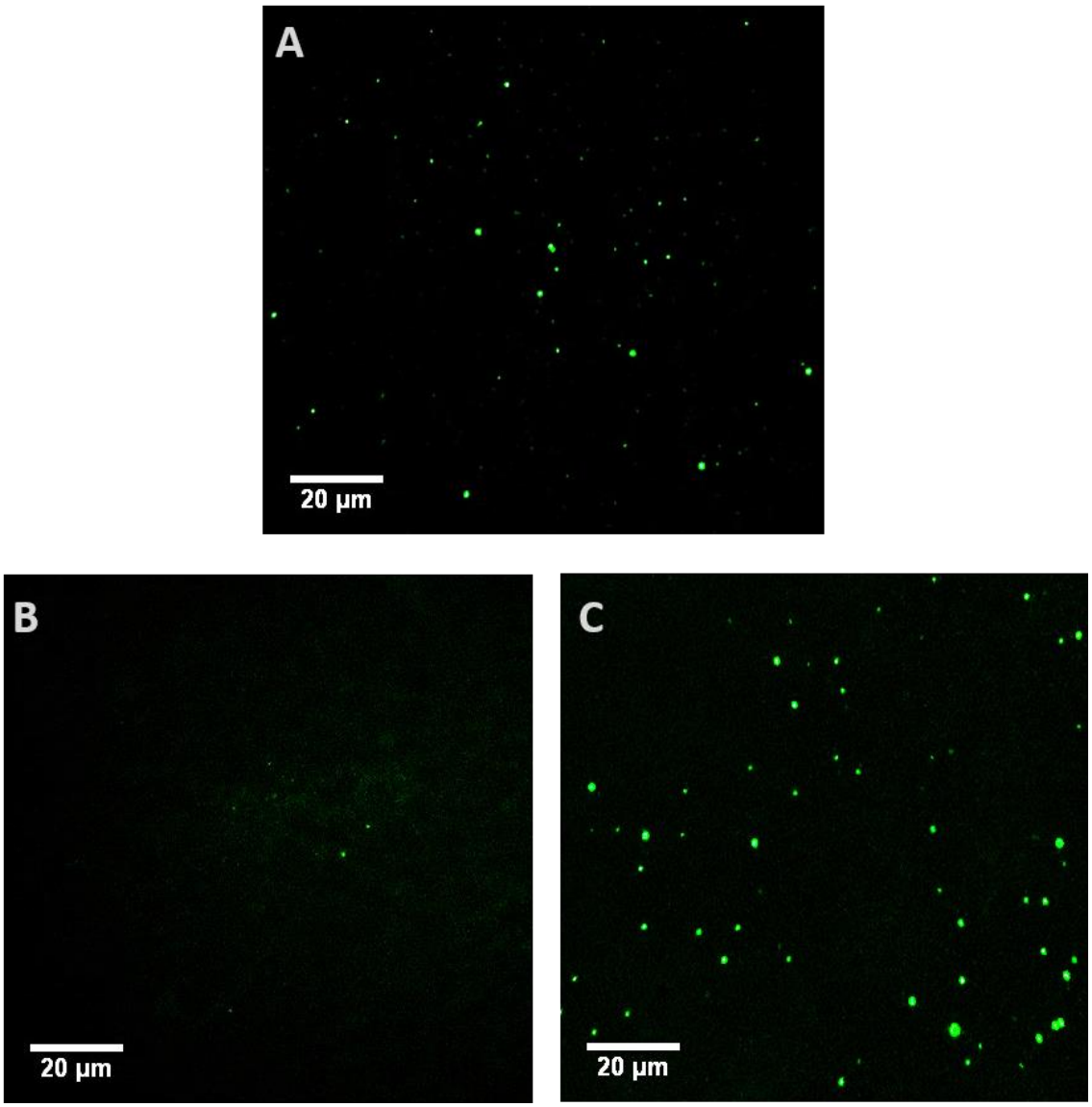

Figure 2.6 Study the specific binding of the receptors inserted in a vesicle with the corresponding antibody tethered on the surface. The presence of vesicles of CFTR-GFP immobilized with the Anti-GFP antibody (A) but absent with the anti-HA antibody (B), indicated that vesicles bound specifically to the glass substrate. The spots seen with CFTR-HA EGFR-GFP immobilized with anti-HA (C) indicated that the anti-HA antibodies were active but did not bind with GFPs in image (B). Thus, our observation was limited to the specific binding of the receptors on the surface of the glass substrate.

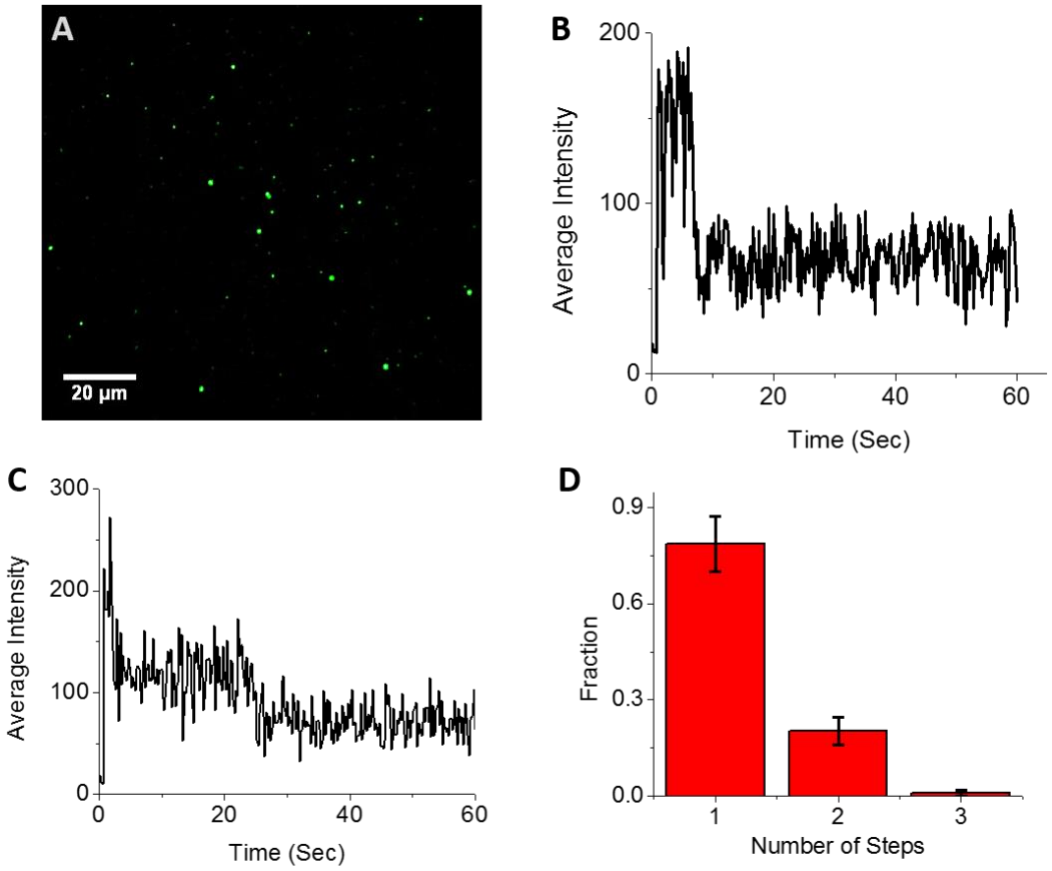

Figure 2.7 Determining if cell-derived vesicles can be utilized to study oligomerization of membrane proteins. (A) A representative image from a movie acquired by immobilizing CFTR-GFP containing vesicles on a coverslip. (B) A representative photobleaching step of CFTR-GFP immobilized on a surface coated with the anti-GFP antibody. The single step photobleaching indicates the presence of a single receptor in the vesicle. (C) Another representative photobleaching event of the CFTR-GFP in a different vesicle; this graph indicated the presence of two CFTR proteins in a vesicle. (D) The data shows that about 80% of the vesicles have single CFTR proteins whereas the remaining 20% of the vesicles contain multiple CFTR proteins.

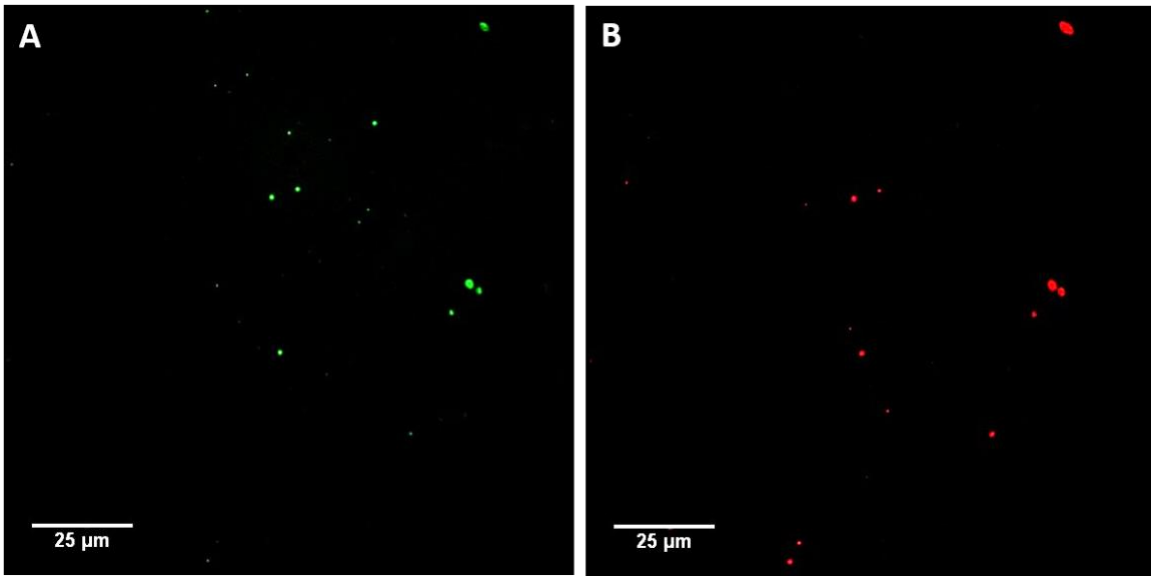

Figure 2.8 Study the functional activity of α7-GFP β2-Flag receptors inserted in the vesicles to probe the fully functional assembly of the receptors. (A) A representative image acquired exciting GFP only, which indicates the positions of receptors. Then acetylcholine and calcium ions are added to open the ion channel and allow calcium ions to enter into the vesicles. Calcium ions bind with Fluo-8 dyes to activate it. (B) A representative image captured exciting the activated Fluo-8 dyes. Image analysis indicates about 60% of the receptors are functional.

vesicles showing functional assembly. It was found that about 60% of the vesicles showed functional activity. Since, to activate Fluo-8 AM dye inside a vesicle, it is necessary to have an esterase inside the vesicle, this might have limited the percentage of receptors showing the functional activity.

2.3.3 Determination of the Stoichiometry of α3β4

The α3β4 nicotinic receptor is expressed in the central and peripheral nervous system. This receptor is a hetero-pentameric receptor and can be assembled into two different stoichiometric isoforms. One isoform can have two α3 subunits and three β4 subunits while other isoform can have three α3 subunits and two β4 subunits. In order to determine the predominant stoichiometry of this receptor, we expressed HEK293T cells with α3-GFP β4wt, and α3-wt β4-GFP separately and then generated vesicles. An anti-GFP antibody coated glass bottom dish was used to immobilize the vesicles (Figure 2.9). The surface tethered receptors were visualized by exciting them with a 488 nm laser with corresponding dichroic and filter under TIRF condition. Vesicles with α3-GFP β4wt showed single, double and triple photobleaching steps of GFP molecules. But the number of two bleaching steps is prevalent which indicates the primary existence of two α3-subunits into the protein complex (Figure 2.9). As nicotinic receptors are pentamers, this result also indicates the predominate existence of three β4-subunits in the complex. Therefore, the predominate stoichiometry of α3β4 nicotinic receptor is $(\alpha3)_2(\beta4)_3$. When movies corresponding to the vesicles with α3-wt β4-GFP were analyzed, it also agreed with the result of α3-GFP β4-wt displaying $(\alpha3)_2(\beta4)_3$ as a predominate stoichiometry of α3β4 nicotinic receptor. The α3β4 nicotinic receptor cannot have one alpha or beta subunit, but

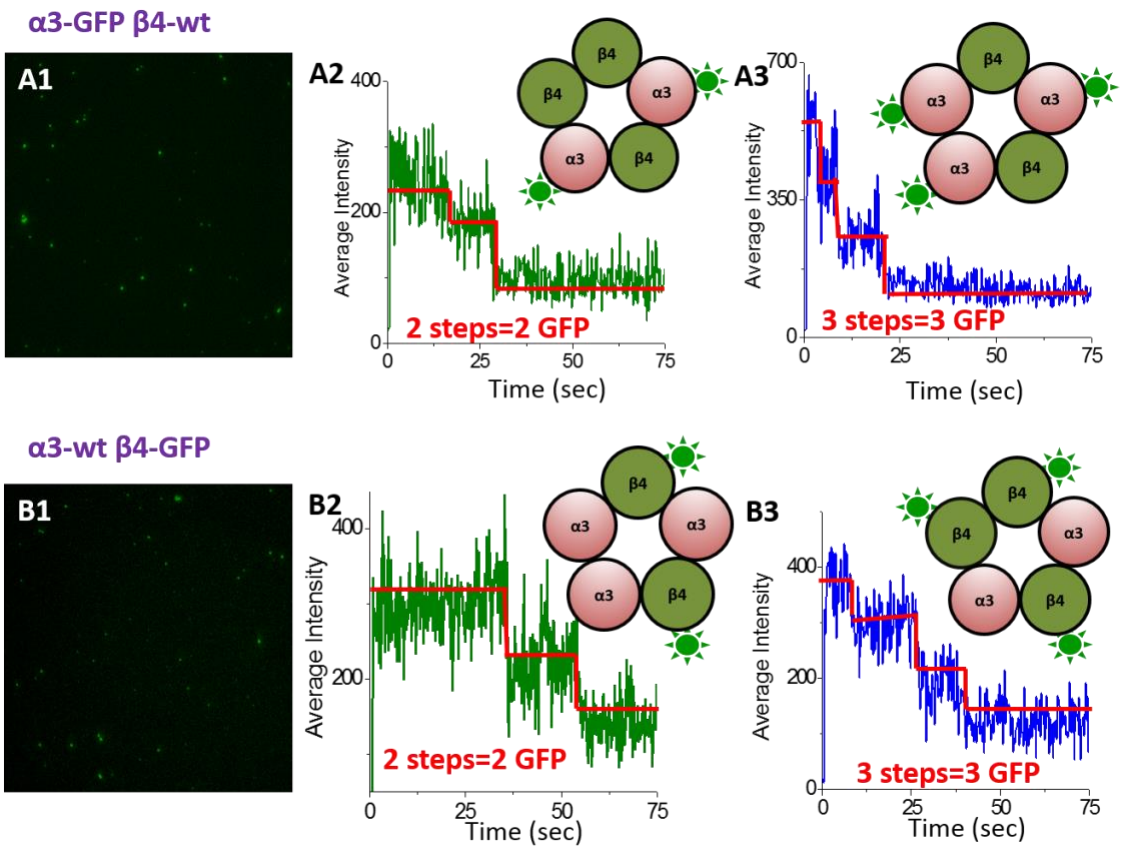

Figure 2.9 Determination of the stoichiometry of α3β4 nicotinic receptors. A1-A3 show data for α3-GFP β4-wt and B1-B3 for α3-wt β4-GFP containing vesicles. (A1) A representative image of the immobilized vesicles with TIRF excitation where every spot represents a molecule. (A2 and A3) Two representative bleaching steps of GFP indicate the presence of two and three α3 subunits respectively. (B1) A typical field of view of the surface tethered α3-wt β4-GFP containing vesicles under TIRF microscopy. (B2 and B3) Two representative bleaching steps indicate the presence of two and three β4 subunits.

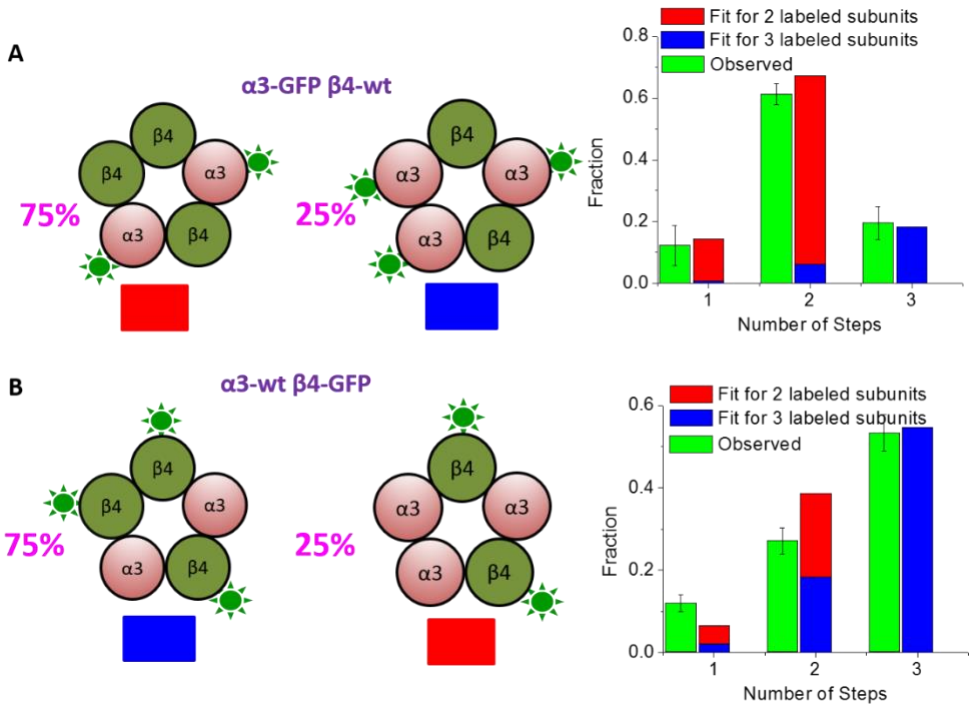

Figure 2.10 Determination of the stoichiometry of α3β4 nicotinic receptors using binomial distribution. (A) It indicates that, with 90% probability of observing GFP, if we provide 75% weight to the probability of observing of two α3-GFP subunits and 25% to that of three α3-GFP subunits in the mixed stoichiometry of α3-GFP β4wt receptor, the experimental and theoretical data are in a good agreement. (B) This represents the same results with α3-wt β4-GFP receptor where 75% weight is provided to the probability of observing three β4-GFP (complementary to two α3 subunits) and 25% to that of two β4-GFP (complementary to three α3 subunits). Therefore the predominate stoichiometry have two α3 subunits and three β4 subunits. The error bars for the subunit distribution are based on counting events and are calculated as the square root of the counts.

a number of single photobleaching step was observed. This might arise because of the possibility of the nonzero probability of GFP being non-fluorescent. Thus, it is necessary to carry out a statistical analysis to conclude the predominate stoichiometry of the receptor.

We determined two theoretical distributions of possible one, two and three photobleaching events from two GFP and three GFP molecules using Binomial distribution and a probability of 90% to a GFP being observable (see details in 2.2.14). These two binomial distributions were weighted to obtain a theoretical or expected distribution which was compared with observed distribution. When we weighted the probability distribution of two α3-GFP by 75% and three α3-GFP by 25, the theoretical data showed a good agreement with observed data of %, for α3-GFP β4wt receptors (Figure 2.10A). Later, with α3-wt β4-GFP receptors, we assigned the same probability of observing a GFP, and weighted the probability of observing three β4-GFP by 75% and two β4-GFP by 25%. This result also shows a good agreement with the experimental data. Hence, whether GFP was incorporated into the alpha or beta subunit of the α3β4 nicotinic receptor, 75% of the receptors had two alpha subunits and three beta subunits, i.e. $(\alpha3)_2(\beta4)_3$ stoichiometry; while remaining showed $(\alpha3)_3(\beta4)_2$.

While the results obtained from the alternate subunit labeling with GFP were consistent, a simultaneous counting of all 5 subunits within the same receptor could provide definitive evidence of the accurate stoichiometry. So, we have derived vesicles from HEK293T cells with α3-mCherry β4-GFP receptor and immobilized on the surface of a glass substrate though anti-GFP antibody. The 488nm and 561nm lasers and corresponding filters and dichroics were employed to visualize GFP and mCherry respectively

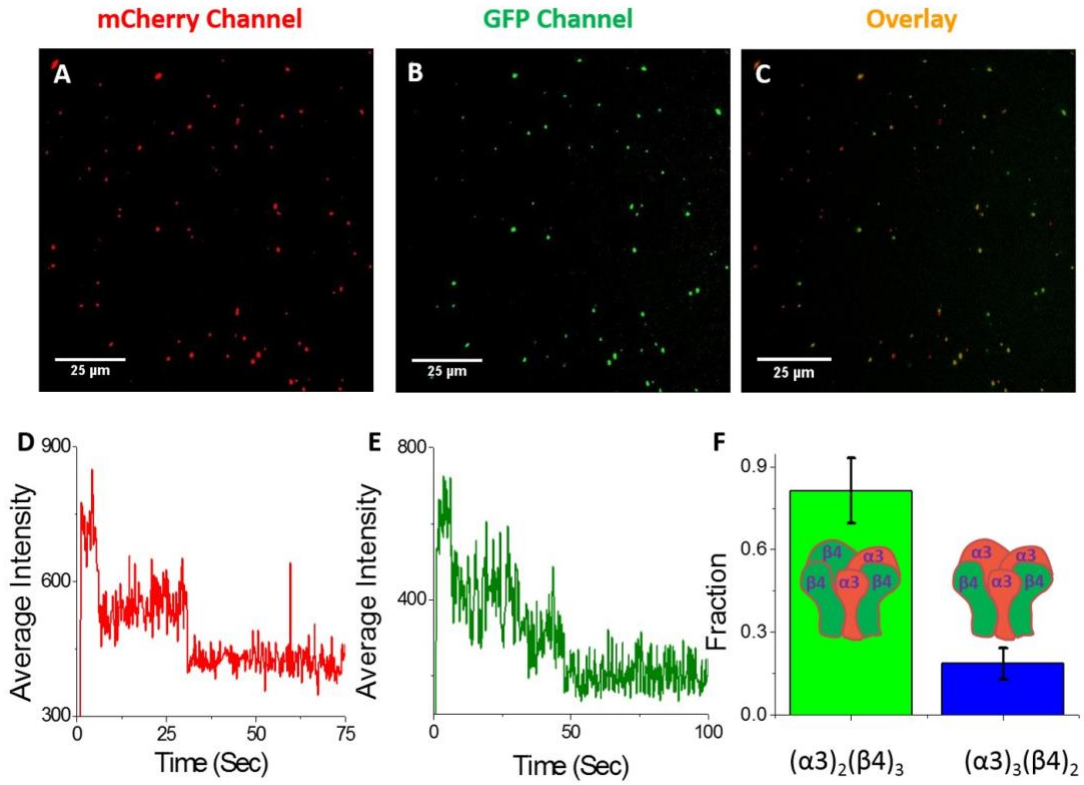

Figure 2.11 Determination of the stoichiometry of α3β4 nicotinic receptors by counting both of the subunits simultaneously. The dual fluorophore labeled receptor (α3-mCherry β4-GFP) containing vesicles were excited to visualize mCherry (A), and GFP molecules (B). (C) An image obtained by overlaying the GFP and the mCherry channel on the same field of view. (D and E) Two are representative bleaching steps of mCherry and GFP labeled subunits, respectively, from the same receptor showing the presence of two α3 subunits and three β4 subunits. (F) We counted only those events that showed a total of 5 bleaching steps (two mCherry & three GFP or three mCherry & two GFP). The result indicates that α3β4 primarily assembles as $(\alpha3)_2$ $(\beta4)_3$.

(Figure 2.11). Moves were taken with 561 nm laser first and then 488 nm laser on the same field of views. Time traces were obtained from the overlayed spots and subjected to analysis. We counted only those spots that showed a total of 5 bleaching steps (2-mCherry & 3-GFP) or (3-mCherry & 2-GFP). This dual color experiment also agreed with that of single color experiment that the $(\alpha 3)_2(\beta 4)_3$ is the predominate stoichiometry of the α3β4 nicotinic receptor.

2.3.4 Study of Ligand Receptor Interaction

In order to show the diversity of the application of vesicles generated from the live cells, we have shown that these vesicles can be employed to study ligand receptor interactions. We have generated EGFR-GFP containing vesicles from HEK293T cells. FCS data shows that a freely moving EGFR-GFP containing vesicles (by exciting GFPs at 488 nm) have a mean diffusion time of around 33 ms while that of an EGF conjugated with tetramethylrhodamine (EGF-TMR) (by exciting TMR with 561nm laser) is about 1.0 ms (Figure 2.12). When EGFR-GFP containing vesicles were mixed with EGF-TMR, we obtained two diffusion times by exciting TMR with 561nm laser. One diffusion time was at 1 ms, corresponding to unbound ligands and another was at 33 ms indicating the interaction of the ligand with the receptor inserted in the vesicle.

We have also examined ligand receptor interactions with immobilized receptors inserted in vesicles. Vesicles were generated by co-transfecting HEK293T cells with EGFR-GFP and CFTR-HA DNA plasmids. Vesicles (~100 µl) were mixed with EGF-TMR (so that final concentration is 1 µM); the mixed solution was dialyzed with a Slide-

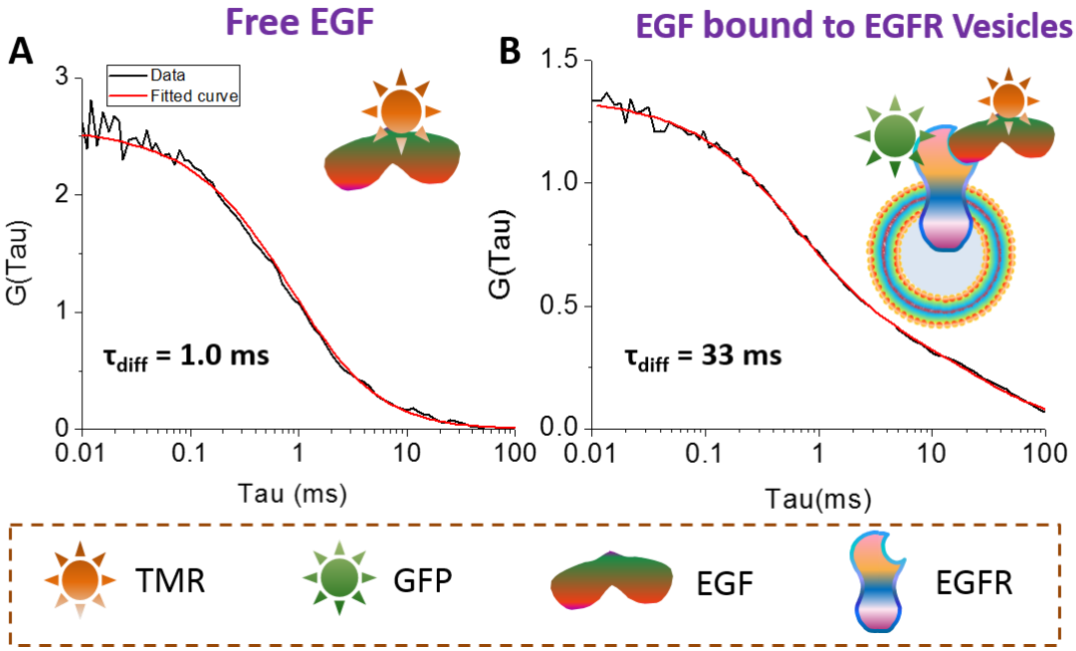

Figure 2.12 Application of cell-derived vesicles for the study of ligand receptor interaction. (A) FCS curve of an epidermal growth factor (EGF) conjugated with tetramethylrhodamine (TMR) showed a diffusion time of 1 millisecond. (B) When EGF-TMR was mixed with EGFR-GFP containing vesicles and FCS data was recorded by exciting TMR, it shows a diffusion time of ~33 milliseconds along with a diffusion time of unbound ligands (1 ms). The shift in the diffusion time clearly indicates the biding of the ligands with receptors.

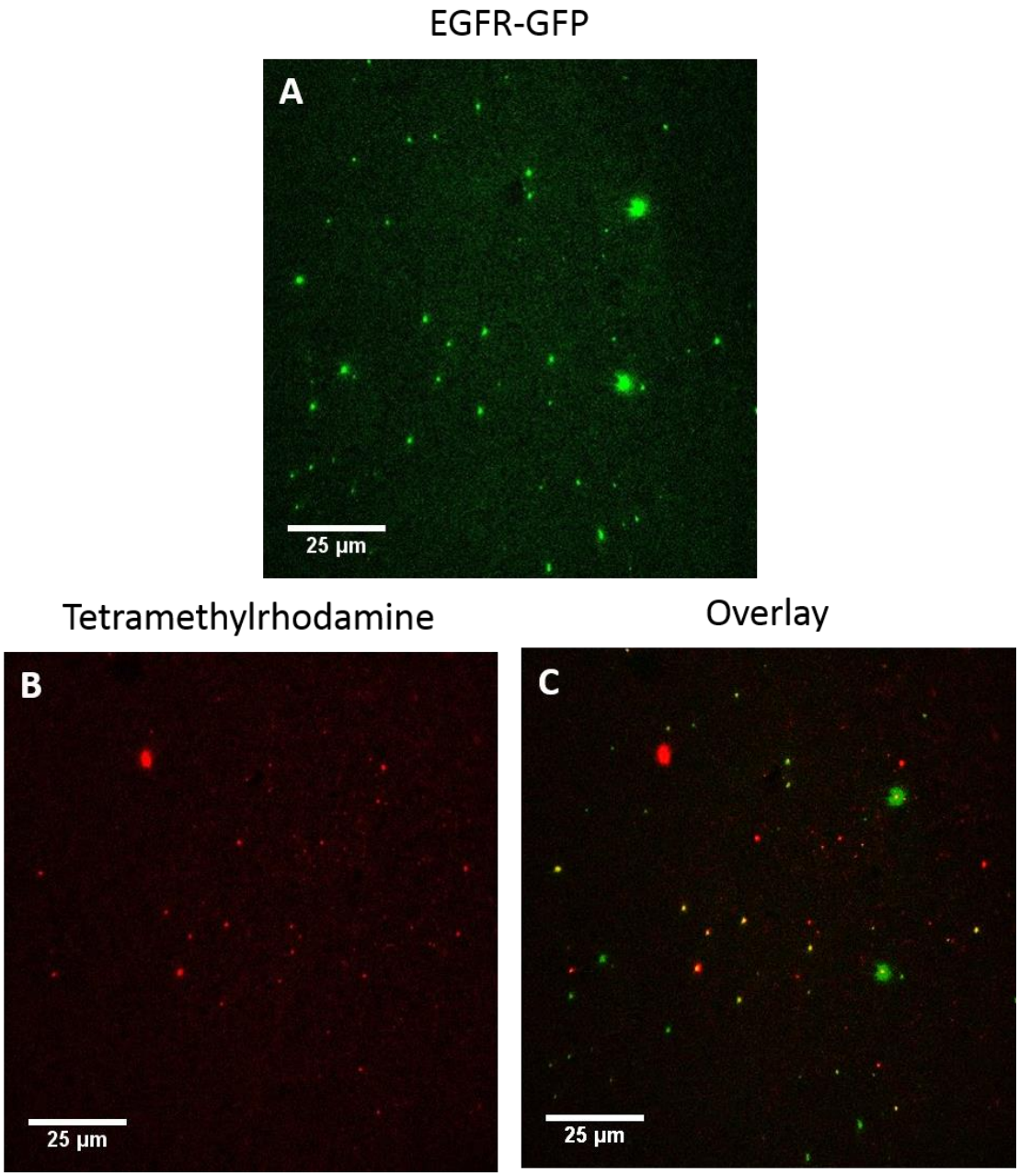

Figure 2.13 Study ligand receptor interaction on the surface of a glass substrate. EGFR-GFP and CFTR-HA containing vesicles were immobilized with an anti-HA antibody coated coverslip. (A) Receptors (EGFR) were visualized by exciting GFPs; (B) Ligands (EGFs) were located by exciting tetramethylrhodamine (TMR) as EGF was conjugated with TMR. (C) Overlay image indicates ligands and receptors were present on the same spots for the same field of view.

A-Layer MINI Dialysis Devices, 10K MWCO (Thermo Scientific) in 250 ml PBS buffer for 30 min at room temperature on top of a magnetic stirrer. Then, the buffer was replaced with a new 250 ml PBS buffer and dialysis was carried out. Buffer was changed in every 30 minutes for a total 8 times to make sure all unbound dyes was removed. An anti-HA antibody coated coverslip was employed to immobilize 100 –fold diluted purified vesicles; 488 nm and 561 laser beams and corresponding filter and dichroic sets were used to visualized receptors (through GFP of EGFR) and ligand (by locating TMR of EGF) respectively (Figure 2.13). The overlay of receptor and ligand of the same field of view show the specific binding of ligand and receptor.

2.4 Conclusion

Single molecule studies of membrane receptors are limited to live cells because of the high expression level and movement along the membrane of the membrane proteins. Isolation of the receptors in artificial membrane bilayers requires transferring proteins from a cell membrane to the artificial membrane that includes an intermediate step of dissolution of the protein in a detergent solution. This step endangers the loss of the functional integrity of the large membrane receptor, thus hindering the application of the single molecular studies of the receptors. Here we presented a new method where a receptor was isolated into a cell derived vesicle which eliminated the intermediate step and kept the proteins always in its physiological environment. Our preparation has shown to have a vesicle of about 200 nm in diameter; and about 15% of the vesicles have receptors in them. Therefore, about 98% of the vesicles will have single receptors. These isolated receptors are assembled intact in the vesicles, which has been validated using the characteristic calcium

flux of α7β2 nicotinic receptors. We were able to isolate α3β4 nicotinic receptors in the cell derived vesicles and found a mixed stoichiometry where predominate assembly is $(\alpha 3)_2(\beta 4)_3$. This result was verified with alternatively single subunit labeling and with simultaneously both subunit labeling with fluorescent proteins. We have also shown that ligand receptor binding can also be studied with the generated vesicles. We have employed vesicles inserted with EGFR and corresponding ligands – EGF – to demonstrate the ligand receptor interaction study in solution (with FCS) and on the surface of a glass substrate (with TIRF). We strongly believe that these vesicles will enable isolation of all sorts of transmembrane proteins to carry out single molecule studies.

Chapter 3

Organelle-Specific Single-Molecule Imaging of α4β2 Nicotinic Receptors to Understand their Assembly and Trafficking

Contributions: The work presented in this chapter was accomplished in collaboration with Dr. Ashley M. Loe. She completed the work related to the isolation and characterization of ER and plasma membrane originated vesicles as well as single molecule studies of the ER and Plasma membrane originated vesicles generated with and without the presence of nicotine. She also completed the SEP based studies of α4β2 to understand ligand induced upregulation. I did the experiments to understand the assembly of α4β2 nicotinic receptors from whole cell samples with various nicotinic receptor ligands and the biased transfection based studies from the ER and plasma membrane originated vesicles as well the whole cell originated vesicles. Additionally, I wrote the software packages to collect time traces, to count number of photobleaching events, and to fit data to determine the ratio of stoichiometric assemblies of nicotinic receptors. Source code of the software packages can be found in the Appendix.

3.1 Introduction

Nicotinic acetylcholine receptors (nAChRs) assemble in the endoplasmic reticulum (ER) and traffic to the cell surface as pentamers composed of alpha (α2-α10) and beta (β2-β4) subunits (82,221-224). Many subtypes of these receptors can assemble with varying ratios of subunits, giving rise to multiple stoichiometries that exhibit different subcellular localization and functional properties (225-228). In addition to the endogenous neurotransmitter, acetylcholine, nicotine also binds to and activates these receptors. It has been shown that nicotine influences the trafficking of nAChRs resulting in increased expression on the cell surface (141,229-232). It has also been proposed that nicotine and other ligands alter the assembly of the α4β2 nicotinic receptor resulting in a decrease in the low sensitivity stoichiometry, $(\alpha4)_3(\beta2)_2$, and an increase in the high sensitivity stoichiometry, $(\alpha4)_2(\beta2)_3$ (10,140,227,233,234). Until now, no techniques were capable of specifically determining the stoichiometry of receptors in the ER versus those on the plasma membrane. We developed a single molecule technique that allowed us to differentiate between receptors localized in the ER and plasma membrane to quantify the stoichiometry of individual receptors (Figure 3.1).

Single molecule fluorescence measurements are widely utilized to investigate protein dynamics including the detection of conformational changes, stoichiometry and protein mobility (10,28,215). Experiments with this level of detail are only feasible after purification of the protein from its physiological cellular environment (235,236). This type of purification is not achievable for many types of membrane proteins, which lose their structural and functional integrity when removed from their native cellular environment.

This restricts single molecule studies to a small number of membrane proteins. Other techniques, such as single-molecule pull-down, have also been used to capture proteins from cellular systems extending single molecule studies to a wider range of proteins (28,237,238). Still, one of the primary limitations to applying these current approaches to membrane receptors is that organelle-specific information is lost during protein purification. Membrane proteins are synthesized in the ER and then trafficked to the cell surface through the secretory pathway. Assembly of individual subunits that comprise oligomeric membrane proteins also takes place in the ER prior to transport to the cell surface. Understanding drug induced changes in the distribution of protein isoforms between the ER and plasma membrane is vital to determining how moderately different structural properties can vastly impact functional properties. Isolation of individual membrane receptors in cell-derived nanoscale vesicles composed of original membranes enables the separation of receptors based on organelle (239). Investigation of oligomeric proteins from specific organelles at a single molecule level provides a way to distinguish between different structural and functional populations of these proteins, allowing the effect of changes in assembly on protein trafficking to be directly studied.

Here we report a novel approach that enables us to perform organelle-specific single molecule studies of membrane proteins. We can effectively select populations from the ER and the plasma membrane to quantify properties such as the distribution of stoichiometric assemblies of oligomeric proteins. We applied this technique to study nicotine-induced changes in the assembly of α4β2 nAChRs in the ER and changes in trafficking to the cell surface.

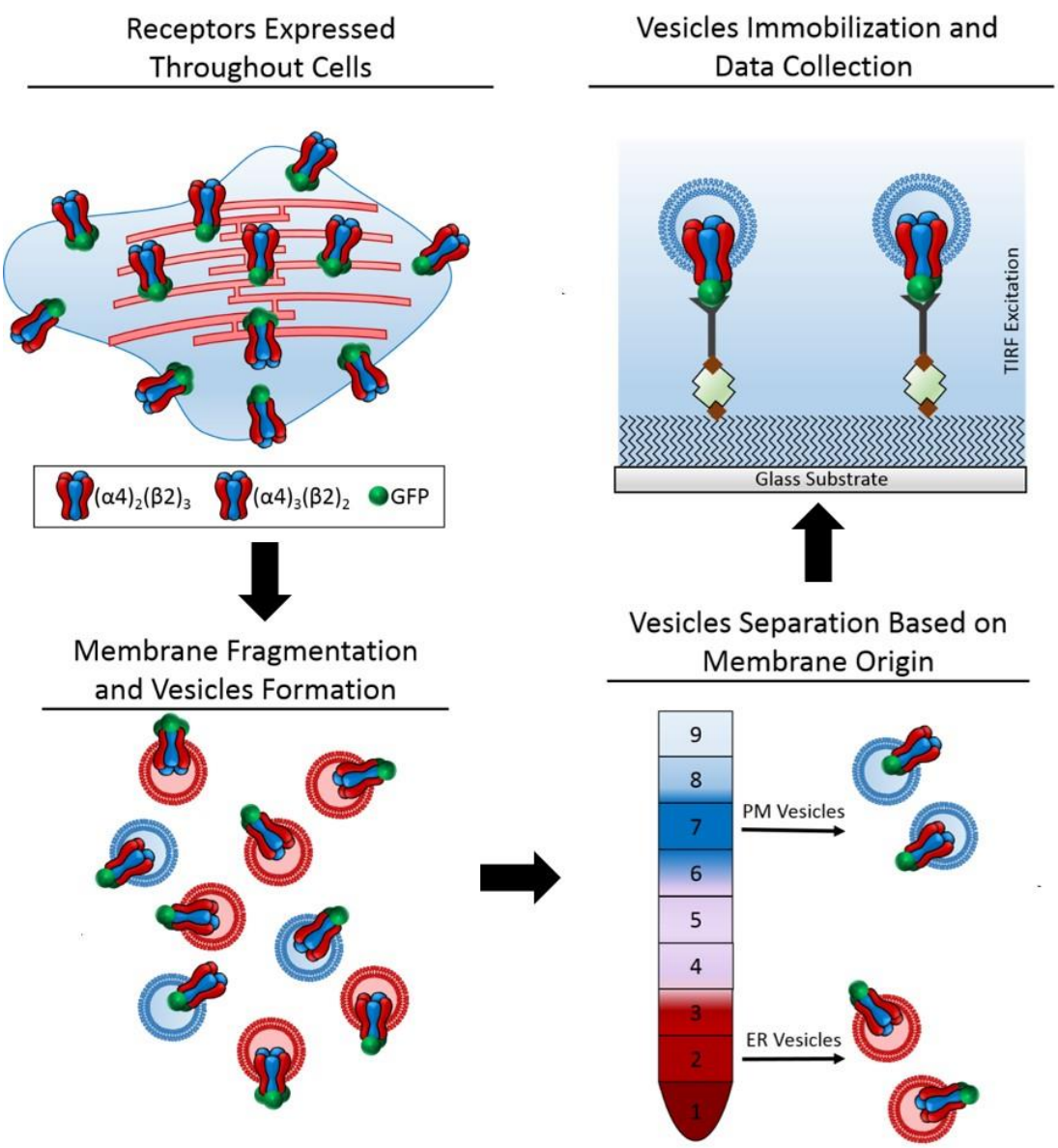

Figure 3.1 Schematic showing the generation of organelle-specific nanovesicles containing a single nAChR. Cells expressing fluorescently labeled membrane receptors are expressed throughout the ER and on the plasma membrane. Nitrogen cavitation is used to fragment the cells forming small membrane domains from cellular organelles. These membrane domains spontaneously form nanoscale vesicles. The domains and subsequent vesicles are small enough that there is a low probability of more than one receptor being encapsulated. The resulting vesicles have the same membrane properties as the organelle of origin, thus maintaining a physiological environment. Differences in the densities between the organelle membranes are used to separate them via gradient centrifugation. Vesicles are isolated on glass substrates for TIRF imaging

3.2 Experimental Procedures

3.2.1 Plasmid Construct

Plasmid constructs for fluorescently labeled nicotinic receptors (SEP and GFP) were generated as previously reported (239). The α4 –SEP construct was made by fusing the DNA sequence of super ecliptic pHluorin (SEP) to the 3' end of the DNA sequence of the α4 subunit. GFP constructs were made by inserting the label between the M3 and M4 transmembrane segments of the α4 subunit. Both constructs have been shown to produce functional receptors in previous studies (239-241).

3.2.2 Cell Culture

Undifferentiated mouse neuroblastoma 2a (N2a) cells were employed to study the trafficking of α4β2 nAChRs. N2a cells were cultured and maintained with an N2a growth media (equal volume mixture of DMEM and OptiMEM supplemented with 10% fetal bovine serum and 1% penicillin and streptomycin) at 37 °C with 5% CO_2 in a humidified incubator. Approximately 90,000 N2a cells were plated on a poly-D-lysine coated glass bottom dish (35 mm in diameter, Cell E&G, San Diego, CA). The coated dish was prepared by incubating it with 0.1% poly-D-lysine in sterile deionized water at 37 °C for 1 hour. The unbound poly-D-lysine was removed by rinsing with sterile deionized water, and the dish was dried for 2 hours in a biosafety hood. After 16-24 hours, the N2a cells were transfected with 500 ng of each α4-SEP and β2-wt plasmids with 2 μl Lipofectamine 2000 as described previously (239). Briefly, cells were transfected for 24 hours, followed by a 24-hour incubation in growth media prior to imaging. Transfection mix was prepared by incubating

a mixture of 250 µl OptiMEM and 2 µl Lipofectamine 2000 transfection agent for 5 minutes at room temperature, followed by a 25-minute RT incubation upon combination with a mixture of 250 µl OptiMEM and 500 ng of each plasmid DNA. The 500 µl of transfection mix was added to pre-plated cells in 1.5 mL OptiMEM. After 24 hours, transfection media was replaced with N2a growth media for an additional 24-hour incubation. Transfected cells were imaged 48 hours after initial transfection. When applicable, 500 nM of each nicotinic ligand, (−)-nicotine hydrogen tartrate salt (>98%), bupropion hydrochloride (>98%), varenicline tartrate, or (−)-Cytisine (>99%), was added to the transfection media and replenished later in the growth media. Transfection efficiency was generally 80% and was not significantly altered by the presence of any of the ligands.

3.2.3 Total Internal Reflection Fluorescence

The total internal reflection fluorescence (TIRF) microscope system employed to visualize SEP or GFP molecules was previously described (239,242). Briefly, a 488 nm DPSS laser excitation source was directed toward the back aperture of an objective (60x, 1.49 NA) mounted on an inverted microscope (Olympus IX81). The angle of excitation light was adjusted to obtain total internal reflection through the objective using a stepper motor that translated the beam across the back aperture of the objective. The emission was collected though the objective and a dichroic mirror was used to direct the light to an EMCCD camera (Andor).

3.2.4 Receptor Expression and Distribution

For *in vivo* fluorescence imaging studies, the growth media of the transfected N2a cells was replaced with an extracellular solution (10mM HEPES, 10 mM D-glucose, 150 mM NaCl, 4 mM KCl, 2 mM $MgCl_2$ and 2 mM $CaCl_2$) of pH 7.4. The dish was then mounted on a translational stage and cells were located by exciting the SEP molecules with a 488 nm laser (~1mW) source. Images of cells were captured using an EMCCD camera with a 200 ms exposure time. The extracellular solution was then replaced with an identical solution of pH 5.4, followed by a 10-minute stage-top incubation before capturing images of the same cells. An open source software, ImageJ (NIH, USA) was employed to analyze the images. Background was subtracted using the rolling ball background subtraction with a diameter of 25 pixels. A freehand region of interest (ROI) was drawn around a cell and an intensity based threshold was used to obtain an integrated density for each cell. The integrated density of the cell at pH 5.4 (ER ID) is subtracted from the integrated density of the same cell at pH 7.4 (total ID) to calculate the relative number of receptors on the plasma membrane, or plasma membrane integrated density (PM ID). The percentage of receptors located on the plasma membrane within the TIRF region of excitation (% PM) is calculated by dividing the PM ID by the total ID at pH 7.4, multiplied by 100. Data are reported as mean ± STD.

3.2.5 Nanovesicle Preparation

Human embryonic kidney 293T (HEK-293T) cells were cultured and maintained with a growth media (DMEM supplemented with 10% fetal bovine serum and 1% penicillin and streptomycin) at 37 °C temperature with 5% CO_2 in a humidified incubator.

Three million HEK-293T cells were plated in a matrigel coated T75 flask 16-24 hours prior to transfection. Cells were transfected with 14 μl Lipofectamine 2000 and 3.5μg of each plasmid as previously described (239). For biased expression experiments, a 1:10 transfection ratio of α4-GFP:β2-wt using 1 μg α4-GFP and 10 μg β2-wt was employed. Briefly, a mixture of 250 μl OptiMEM and the above mentioned amount of α4-GFP and β2-wt plasmids was prepared. Separately, 14 μl Lipofectamine 2000 was added to 250 μl OptiMEM and incubated for 5 min at RT before being added to the DNA mixture. This new mixture was incubated at room temperature for 25 minutes. Afterwards, the transfection mixture was added to the flask of HEK-293T cells. The following day, vesicles were prepared from transfected cells as previously described (239). Briefly, cells underwent nitrogen cavitation at 250 psi for 5 minutes while suspended in 5 mL sucrose-HEPES buffer supplemented with a protease inhibitor (250 mM sucrose, 10 mM HEPES, 1 Pierce protease inhibitor mini tablet per 10 mL buffer (ThermoScientific), pH 7.5). Cell lysate was then centrifuged at 4000 x g for 10 minutes. Supernatant was collected and centrifuged at 10,000 x g for 20 minutes. Supernatant was again collected and centrifuged at 100,000 x g for 1 hour. The pellet was resuspended in 800 μl sucrose-HEPES buffer (250 mM sucrose, 10 mM HEPES, pH 7.5). Nanovesicles were stored at -80 ^{o}C until use.

3.2.6 Generation of ER and Plasma Membrane Vesicles

HEK-293T cells were transfected as described above. After transfection for 24 hours, transfection mix was removed and cells were rinsed once with PBS. To generate nanoscale plasma membrane vesicles containing a single nAChR, transfected cells were first swollen for 20 minutes in a hypotonic solution (10 mM NaCl, 10 mM Tris-HCl, 1.5

mM MgCl2, 0.2 mM CaCl2, pH 7.4) at 0 °C. To prepare both plasma membrane and ER derived vesicles, cells were treated with 5 mL 1x versene (Invitrogen), incubated at 37 °C for 5 minutes, and pelleted by centrifugation at 200 x g for 5 minutes, as previously described. The cell pellet was resuspended in 3 mL sucrose buffer plus protease inhibitors (250 mM sucrose, 10 mM HEPES, 1 Pierce protease inhibitor mini tablet per 10 mL buffer (ThermoScientific), pH 7.5) before undergoing nitrogen cavitation in a nitrogen decompressor (Parr Instrument Company, IL, USA). To generate ER nanovesicles, cells were pressurized to ~250 psi for 20 minutes. At this pressure, plasma membrane rupturing is minimal and therefore nanoscale vesicle formation from this organelle is negligible. To generate plasma membrane nanovesicles, cells were pressurized to ~600 psi for 20 minutes. Cell lysate was collected and dispensed onto an OptiPrep gradient.

3.2.7 Separation of Organelle-Specific Vesicles

A 9-fraction OptiPrep (60% (w/v) iodixanol in H_2O, Accurate Chemical & Scientific Corp., NY, USA) gradient was used to purify organelle-specific nanovesicles. Gradient solutions of OptiPrep were prepared by diluting the 60% stock solution to 30%, 20%, and 10% in sucrose-HEPES buffer (250 mM sucrose, 10 mM HEPES, pH 7.5), and stored at 4 ^{o}C. The gradient was prepared in an Ultra-Clear centrifuge tube (Beckman Coulter), with 3 mL of the densest fraction added first. ER or plasma membrane nanovesicles containing cell lysate, based on nitrogen pressure during cavitation, was dispensed on top of the 10% fraction, before centrifugation at 112,000 x g for 1.5 hours. After centrifugation, nine 1-1.5 mL fractions, with density interfaces in the same fraction, were collected using a peristaltic pump. Tubing connected to the pump was vertically

inserted into the centrifuge tube so that the highest density fraction is collected first. After fractionation, OptiPrep was removed from nanovesicles by centrifugation at 10,000 x g for 1 hour.

3.2.8 Western Blot Analysis

Resuspended OptiPrep fractions containing membrane proteins were ran on a prepackaged NuPAGE 4-12% Bis-Tris gel (Life Technologies), followed by transfer to a nitrocellulose membrane. The membrane was first blocked for one hour with a PBST solution (5% non-fat milk, 0.1% Tween in PBS). Primary antibodies specific for calnexin (Santa Cruz, calnexin antibody (H-70): sc-11397) or plasma membrane calcium ATPase (PMCA) (Santa Cruz, PMCA antibody (D-1): sc-271193) were added to the membrane in a 1:1000 dilution and incubated overnight at 4^{o}C. Endogenous calnexin is solely found in the membrane of the ER, while PMCA is expressed on the plasma membrane, thus providing a means to identify fractions that consist of exclusively ER or plasma membranes. After overnight incubation, primary antibodies were removed by four repeated five minute washes with PBST. Secondary rabbit antibody (calnexin) or mouse antibody (PMCA) (Jackson ImmunoResearch) was added in a 1:5000 dilution and incubated for one hour at room temperature, followed by another series of four repeated five minute washes with PBST. Bands were visualized by addition of western blotting substrate for chemiluminescence (Clarity, Bio-Rad) on a Chemi-Doc system (Bio-Rad). In order to validate these results, blots were repeated with a completely different set of antibodies. For primary antibodies, we used 1:2000 diluted rabbit monoclonal anti calnexin (ab92573, Abcam) for ER identification and 1:2000 diluted rabbit monoclonal anti Na K

ATPase (ab76020, Abcam) for plasma membrane identification. In both sets of orthogonal studies the ER and plasma membrane bands matched the expected molecular weights.

3.2.9 Imaging Nanovesicles

A 35 mm glass bottom dish was cleaned by sonicating the dish in 5 M NaOH solution for 30 minutes at 45 °C and then in 0.1 M HCl solution for 30 minutes at 45 °C. The dish was rinsed with water and sprayed with 100% ethanol three times after each step, and then dried using compressed air. Finally, the dishes were treated in an oxygen plasma (21 % oxygen for ~5 minutes). A biotinylated anti-GFP antibody functionalized glass bottom dish was prepared by incubating a cleaned dish at room temperature with 1 mg/ml Silane-PEG-Biotin in 95% Ethanol for 30 minutes, 0.1 mg/ml NeutrAvidin in PBS (1x phosphate buffered saline, pH 7.4) solution for 5 min, and finally 1 μg/ml biotinylated anti GFP antibody in PBS for 15 minutes. Between each of the steps, the dish was rinsed three times with 1X PBS solution. Vesicles were immobilized on the biotinylated anti-GFP antibody functionalized dish by adding 50 to 200 fold diluted vesicles in PBS for 30 minutes at room temperature. The unbound vesicles were removed by rinsing with PBS, and ~1 mL PBS solution was added to the dish. The microscope set-up employed to capture images for SEP based studies was also utilized to obtain movies of about 1000 frames (100 ms exposure time) during 488 nm laser excitation (~3 mW).

3.2.10 Data Analysis

A customized software package was written in Matlab to populate time traces from the movies collected with immobilized vesicles. Briefly, the first 10 frames of a movie

were combined together to make a composite frame which was utilized to find peaks with a user defined threshold level. A 3-pixel by 3-pixel region of interest (ROI) was selected for each peak position to obtain the mean intensity of the ROI. A 5-pixel by 5-pixel ring around the peak was selected, and the mean value of the pixels located on the ring was considered as background which was subtracted from the mean value of the ROI of the corresponding frame to obtain a background subtracted mean intensity of the ROI. Time traces for all peaks were stored in a temporary file. During the initial evaluation, a time trace of the temporary file was accepted if the difference of the mean of the intensities of first 20 frames and last 20 frames was more than twice the standard deviation of last 20 frames. All time traces for the qualified molecules were collected and stored for further analysis.

A photobleaching step was counted only if it lasted at least one second and the intensity levels of a step and the next lower level had a difference of at least twice the standard deviation of the lower level. A time trace was considered to arise from a single molecule if it showed at least one clear bleaching step. Each set of data was independently analyzed at least twice and the results were compared.

3.2.11 Data Fitting

The probability of observing a photobleaching event from a GFP molecule is less than 1. Therefore, a binomial distribution was employed to determine the distribution of the number of photobleaching events observed from a population of GFP labeled receptors.

A general equation for observing k number of photobleaching events from n number of GFP labeled receptors can be written as:

$$\mathbf{F(k;\ n, p)} = \frac{\boldsymbol{n!}}{\boldsymbol{k!(n-k)!}}\boldsymbol{p^k(1-p)^{n-k}} \qquad \textbf{Eq. 3.1}$$

Where p is the probability of observing a photobleaching event from a GFP labeled subunit which has been previously determined as 0.90 (239).

A matrix (M_2) with the probabilities of obtaining 1, 2, and 3 photobleaching events from two GFP containing α4β2 nAChRs (i.e. $(\alpha4\text{-GFP})_2(\beta2\text{-wt})_3$) can be written as:

$$\boldsymbol{M_2} = [\mathbf{F(1;\ 2, 0.9)}, \mathbf{F(2;\ 2, 0.9)}, \mathbf{0}] \qquad \textbf{Eq. 3.2}$$

Similarly, a matrix M_3 containing the probabilities of observing 1, 2, and 3 photobleaching events from three GFP containing α4β2 nAChRs (i.e. $(\alpha4\text{-GFP})_3(\beta2\text{-wt})_2$) can be expressed as:

$$\boldsymbol{M_3} = [\mathbf{F(1;\ 3, 0.9)}, \mathbf{F(2;\ 3, 0.9)}, \mathbf{F(3;\ 3, 0.9)}] \qquad \textbf{Eq. 3.3}$$

Since the probability of obtaining zero photobleaching events from 2 or 3 GFP labeled α4β2 nAChRs can be greater than zero, the probability distributions, M_2 and M_3, were normalized as follow:

$$\boldsymbol{M'_2} = \left[\frac{\mathbf{F(1;\ 2,0.9)}}{\boldsymbol{S_2}}, \frac{\mathbf{F(2;\ 2,0.9)}}{\boldsymbol{S_2}}, \mathbf{0}\right] \qquad \textbf{Eq. 3.4}$$

$$\boldsymbol{M'_3} = \left[\frac{\mathbf{F(1;\ 3,0.9)}}{\boldsymbol{S_3}}, \frac{\mathbf{F(2;\ 3,0.9)}}{\boldsymbol{S_3}}, \frac{\mathbf{F(3;\ 3,0.9)}}{\boldsymbol{S_3}}\right] \qquad \textbf{Eq. 3.5}$$

Where M'_2 and M'_3 are normalized probability distribution matrices corresponding to M_2 and M_3 respectively, S_2=F(1;2,0.90) + F(2;2,0.90) + 0 and , S_3=F(1;3,0.90) + F(2;3,0.90) + F(3;3,0.90).

Since the experimentally observed distribution emerged from a mixture of $(\alpha 4\text{-GFP})_2(\beta 2\text{-wt})_3$ and $(\alpha 4\text{-GFP})_3(\beta 2\text{-wt})_2$ stoichiometries, a theoretical probability distribution (T_{pd}) was computed by providing a weight to each normalized probability matrix:

$$\mathbf{T_{pd} = a_2 \times M'_2 + a_3 \times M'_3} \qquad \textbf{Eq. 3.6}$$

Where, a_2 and a_3 are the weights assigned to M'_2 and M'_3 distributions respectively and $a_2 + a_3 = 1$. Therefore, a_2 and a_3 are the proportions of $(\alpha 4\text{-GFP})_2(\beta 2\text{-wt})_3$ and $(\alpha 4\text{-GFP})_3(\beta 2\text{-wt})_2$ stoichiometries, respectively. This probability distribution (T_{pd}) was multiplied by total number of observed 1, 2, and 3 photobleaching events to generate a theoretical distribution. A Chi-square goodness-of-fit test was employed to compare theoretical and observed distributions. The error bars for subunit distribution are based on counting events and are calculated as the square root of the counts. The values of a_2 and a_3 were iteratively assigned and a Chi-square goodness-of-fit test statistics was calculated for each set of a_2 and a_3. A customized Matlab script was written to calculate a_2 and a_3 from the best Chi-square goodness-of-fit test statistics.

3.3 Results

3.3.1 Ligand Induced Upregulation of α4β2 Receptors

Nicotine and several other nicotinic receptor ligands have been shown to upregulate the number of receptors on the cell surface. It has been hypothesized that this upregulation is connected to changes in receptor stoichiometry (234,243). We first evaluated a series of ligands to determine if they altered the expression and trafficking of α4β2 by using a pH sensitive fluorophore, super ecliptic pHluorin (SEP) (141,244,245). The SEP label was genetically incorporated into the protein sequence of the receptor so that it was on the luminal side of the ER and the extracellular side of the plasma membrane (Figure 3.2A & B). SEP is fluorescent at neutral pH and quenched at acidic pH. Thus, receptors in the ER and plasma membrane will exhibit fluorescence, while receptors in the Golgi and trafficking vesicles are not fluorescent. Using total internal refection fluorescence microscopy, we measured ligand-induced upregulation of α4β2 expression on the plasma membrane as an increase in plasma membrane integrated density (PM ID) and a change in the distribution between the ER and plasma membrane (% PM) as compared to control cells (239,246). Exposure to nicotine or cytisine resulted in a 2.5-fold increase, varenicline yielded a 2-fold increase, and bupropion resulted in a 1.5-fold increase in α4β2 expression on the cell surface (Figure 3.2C). Additionally, the intracellular distribution of α4β2 between the plasma membrane and peripheral ER (% PM) shifted towards the plasma membrane upon exposure to each of these ligands (Fig. 2D). Comparisons between the ligands showed that nicotine provides the highest level of upregulation in terms of expression and distribution towards the plasma membrane.

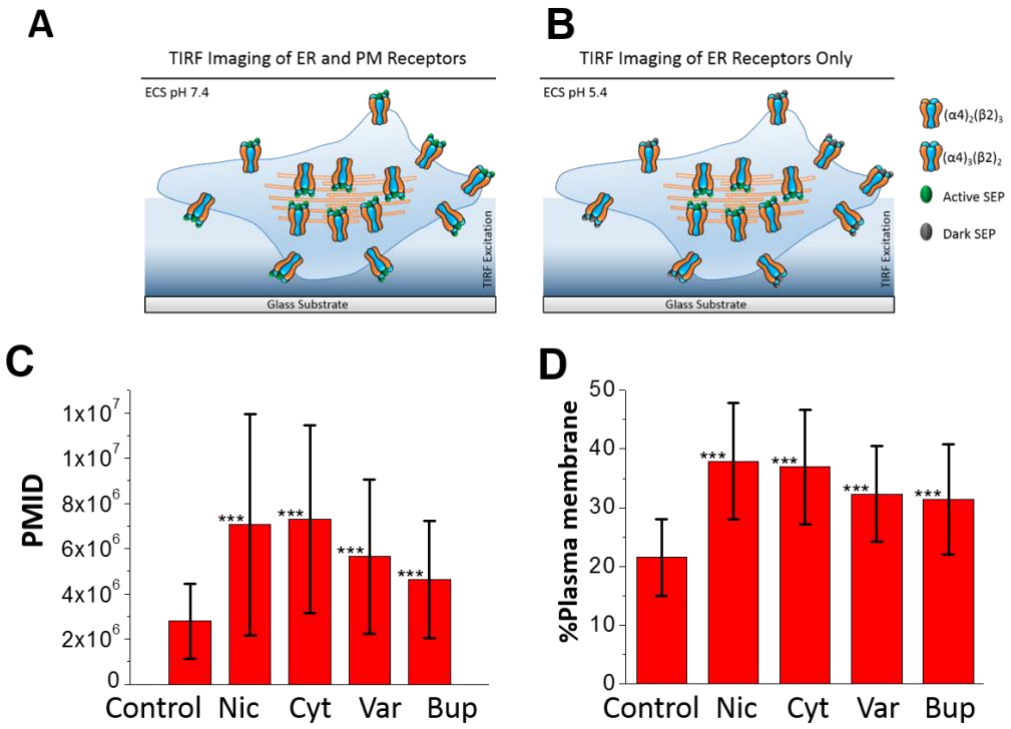

Figure 3.2 Ligand induced upregulation of nAChRs. Schematic showing that TIRF imaging of super ecliptic phluorin (SEP), a pH sensitive analog of GFP, is used to determine the expression and distribution of receptors between the ER and plasma membrane. SEP was genetically encoded into the alpha 4 subunit to generate an α4-SEP construct. The α4-SEP β2-wt nicotinic receptors were expressed in N2a cells and imaged under TIRF. (A) When the pH of the extracellular solution (ECS) was maintained at 7.4, receptors both in the ER and plasma membrane were observable. The observed fluorescence intensity is due to both the ER and plasma membrane receptor populations. (B) When the extracellular solution was replaced with a pH 5.4 solution, all SEP on the plasma membrane transition to a non-fluorescent state and only the receptors within the ER are visible. (C) The integrated density of α4β2 on the plasma membrane increased from approximately 2.5 x 10^6 in the absence of any compound to 7 x 10^6 in the presence of nicotine or cytisine. Varenicline and bupropion both resulted in a 2-fold increase in the integrated density on the plasma membrane demonstrating ligand-induced upregulation. (D) The percentage of the receptors present on the plasma membrane increased from 21.5% for control cells to 30 to 40% for all nicotinic receptor ligands, showing a shift in distribution of receptors toward the plasma membrane. (n = 61, 47, 42, 38, 51) Data are mean values ± S.D.) (***, $p < 0.001$). Integrated density (avg fluorescence intensity x area) is the total gray values background within a region of interest that encompasses a cell. Plasma membrane integrated density (PMID) is obtained by subtracting the integrated density of pH 5 image of a cell from pH 7 image of the same cell.

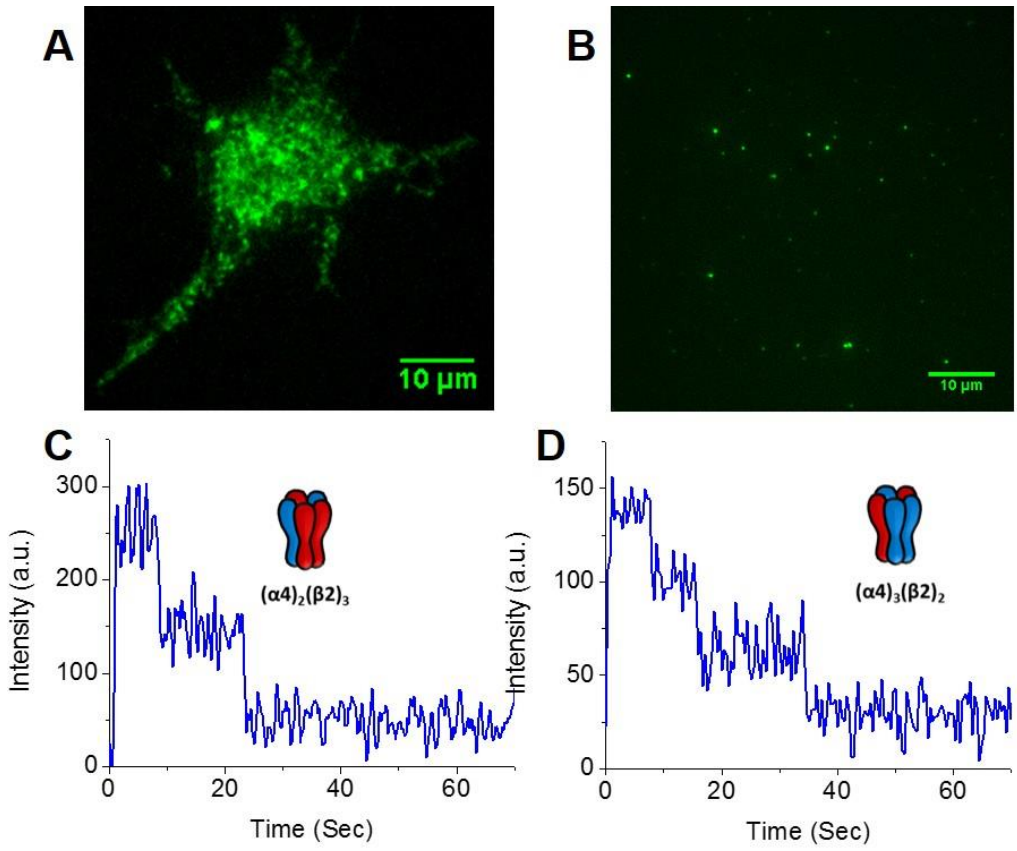

Figure 3.3 Single molecule photobleaching to determine nAChR stoichiometry. (A) Representative mouse N2a cell expressing GFP labeled nicotinic receptors in TIRF. (B) Representative TIRF image of isolated nanovesicles containing individual GFP-labeled receptors on a glass substrate. (C & D) Two representative time traces of two and three photobleaching steps respectively corresponding to two or three GFP-labeled α4 subunits respectively.

3.3.2 Ligand Induced Changes in α4β2 Assembly

In order to determine if ligands that upregulated nicotinic receptors also changed their assembly, we then generated whole-cell nanovesicles from cells expressing α4GFP β2wt nAChRs (Figure 3.3A) and examined the distribution of the two possible α4β2 isoforms, $(\alpha4GFP)_2(\beta2)_3$ and $(\alpha4GFP)_3(\beta2)_2$. Nanovesicles were derived from cells both in the presence and absence of the nicotinic ligands. Single receptors were then isolated into membrane derived vesicles via nitrogen cavitation, immobilized on a glass surface and imaged using total internal reflection fluorescence microscopy (TIRFM). A representative image of isolated vesicles is shown in Figure 3.3B. We then determined the number of photobleaching events (214,217,218) from the intensity time traces recorded for the fluorescence of each nanovesicles (Figure 3.3C & D). The assignment of the stoichiometric distribution is complicated by the fact that a small fraction of GFP exists in a non-fluorescent state (239,247). Additionally, the observed distribution of photobleaching events arises from a combination of $(\alpha4GFP)_2(\beta2)_3$ and $(\alpha4GFP)_3(\beta2)_2$ stoichiometries. To account for these factors, the observed distribution was fit to two binomial distributions, corresponding to the distributions of the photobleaching events of $(\alpha4GFP)_2(\beta2)_3$ and $(\alpha4GFP)_3(\beta2)_2$. They were weighted iteratively to determine the contribution of each stoichiometry. Results from these unsorted vesicles provided a measure of the whole cell distribution of α4β2 stoichiometries. In the absence of any ligand, we observed a distribution of 41% $(\alpha4)_2(\beta2)_3$ and 59% $(\alpha4)_3(\beta2)_2$ (Figure 3.4A). The presence of nicotine shifted the distribution of stoichiometry to 59% $(\alpha4)_2(\beta2)_3$ and 41% $(\alpha4)_3(\beta2)_2$ (Figure 3.4B). A comparison of all ligands showed that each altered the assembly toward

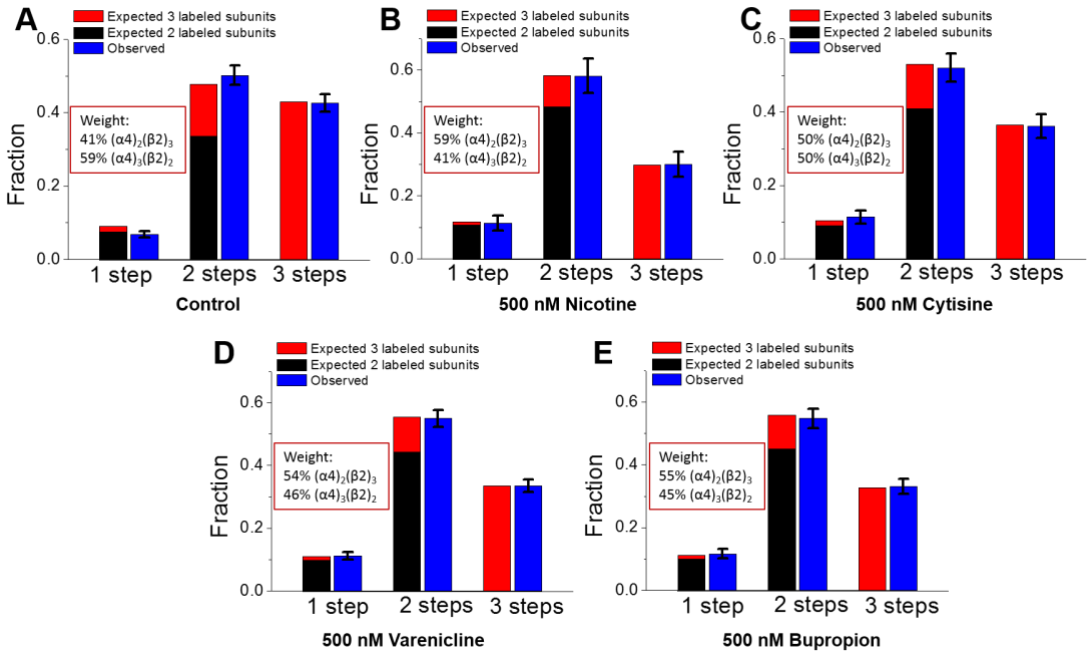

Figure 3.4 Whole cell evaluation of $(\alpha4)_2(\beta2)_3$ versus $(\alpha4)_3(\beta2)_2$assembly upon exposure to nicotinic receptor ligands. Expected distributions of 1, 2, and 3 photobleaching steps were obtained by weighting two binomial distributions. A chi-square goodness of fit test was used to verify expected and observed distributions of two and three GFP labeled α4 subunits. (A) In the absence of a pharmacological agent, the α4β2 population exists as 41% $(\alpha4)_2(\beta2)_3$ and 59% $(\alpha4)_3(\beta2)_2$. (B) 500 nM nicotine alters the ratio of isoforms to 59% $(\alpha4)_2(\beta2)_3$ and 41% $(\alpha4)_3(\beta2)_2$. (C) 500 nM cytisine shifts the stoichiometry to 50% high sensitivity receptors (D) 500 nM varenicline shifts the distribution to 54% high sensitivity receptors. (E) 500 nM bupropion shifts the stoichiometry to 55% high sensitivity receptors, $(\alpha4)_2(\beta2)_3$. The error bars for the subunit distribution are based on counting events and are calculated as the square root of the counts.

Table 3.1 The observed distribution of bleaching steps for whole cell nanovesicles expressing α4β2

	# Vesicles Counted	1 Step	2 Steps	3 Steps	4 Steps
Control	767	53	386	328	54
Nicotine	192	22	112	58	9
Cytisine	357	41	187	130	17
Varenicline	833	94	459	280	30
Bupropion	1089	102	598	389	43

$(\alpha4)_2(\beta2)_3$ with cytisine increasing to 50%, varenicline 54%, and bupropion 55% of the high sensitivity stoichiometry (Figure 3.4C, D, and E). The observed distribution of bleaching steps is shown in Table 3.1.

3.3.3 Organelle-Specific Stoichiometry of α4β2.

We isolated nanovesicles and then sorted them via gradient centrifugation into those derived from the ER and plasma membrane (Figure 3.1). Differences in the density of endogenous ER and plasma membrane allow nanovesicles originating from these organelles to be separated using a density gradient (223). We verified the separation of vesicles using organelle-specific antibodies via western blot analysis. The ER marker, anti-calnexin, was only found in the 3 fractions with the highest density while the PM marker, anti-PMCA (plasma membrane calcium ATPase) or anti-Na^+/K^+ ATPase was found in fractions with the lowest density (Figure 3.5). We utilized this organelle-specific approach to distinguish between changes in receptor stoichiometry during assembly versus altered trafficking. Since nicotinic receptors are synthesized in the ER, differences in assembly are reflected in the stoichiometry of this population. Receptors are trafficked to the cell surface after assembly, thus a change in the stoichiometry on the plasma membrane reflects preferential trafficking or increased stability on the cell surface. We performed separate single molecule studies on both the ER and plasma membrane specific nanovesicles. Single molecule photobleaching analysis relies on the observation of single step bleaching events of GFP where each bleaching event corresponds to a single subunit. Our results using GFP labeled alpha subunits showed the predominately-expressed stoichiometry of α4β2 nAChRs depends on the subcellular region. Receptors encapsulated in nanovesicles

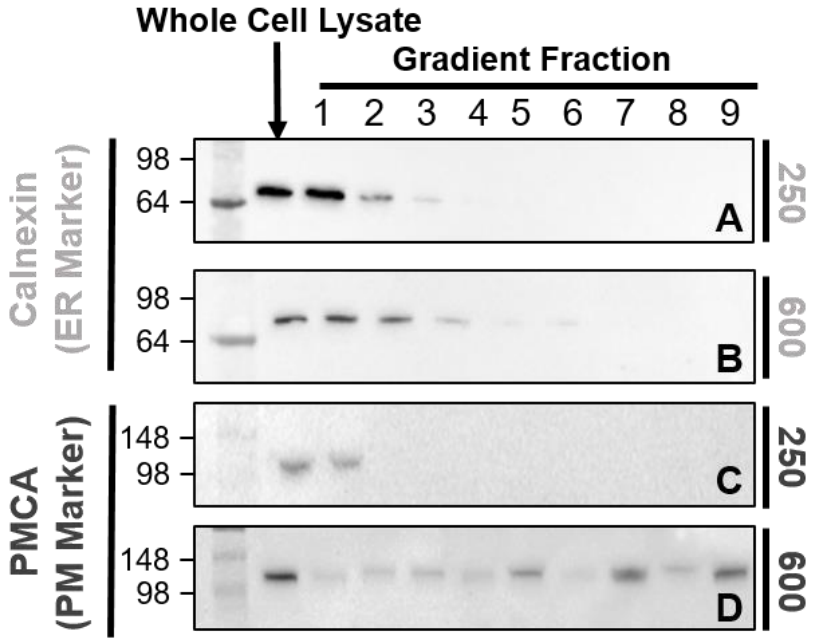

Figure 3.5 Western blots verifying separation of ER and plasma membrane derived nanovesicles. Anti-calnexin was used as an ER marker to identify nanovesicles originating from the ER. Anti-PMCA (plasma membrane calcium ATPase) was used to detect nanovesicles formed from the plasma membrane. Calnexin is detected in higher density fractions when vesicles are formed at 250 psi (A) and at 600 psi (B). Minimal PMCA is detected at fragmentation of 250 psi (C), but are localized to lower density fractions upon swelling with a hypotonic solution and a higher cavitation pressure of 600 psi (D). ER specific nanovesicles are collected from fraction 2 after 250 psi. Plasma membrane specific nanovesicles are collected from fraction 7 after formation at 600 psi.

Table 3.2 The observed distribution of bleaching steps for organelle-specific nanovesicles expressing α4β2

	#Vesicles Counted	1 Step	2 Steps	3 Steps	4 Steps
ER No Drug	458	26	199	233	14
PM No Drug	545	26	199	233	1
ER + 500 nm Nic	465	40	273	152	27
PM + 500 nM Nic	883	100	592	191	18

derived from the ER show that in the absence of nicotine, α4β2 nAChRs predominately assemble with the low sensitivity stoichiometry of $(\alpha4)_3(\beta2)_2$. The distribution of photobleaching events from the ER resident α4β2 nAChRs fit to a theoretical distribution weighted for 30% $(\alpha4)_2(\beta2)_3$ and 70% $(\alpha4)_3(\beta2)_2$ (Figure 3.6A). Single molecule analysis of isolated α4β2 nAChRs encapsulated in nanovesicles derived from the plasma membrane had a higher fraction of the high sensitivity isoform. This fit to 54% $(\alpha4)_2(\beta2)_3$ and 46% $(\alpha4)_3(\beta2)_2$ is shifted from the distribution observed in the ER suggesting that the two isoforms traffic to the cell surface with different efficiencies (Figure 3.6B). The observed distribution of bleaching steps is shown in Table 3.2.

3.3.4 Nicotine changes the stoichiometry of α4β2 in the ER

We next prepared ER and plasma membrane specific vesicles from cells expressing α4β2 in the presence of 500 nM nicotine. We observed a clear nicotine-induced shift in the stoichiometry of ER resident α4β2 receptors (Figure 3.6C). When nicotine was present, single molecule bleaching step analysis showed the majority of endoplasmic α4β2 assembled as the high sensitivity isoform, fitting a 55% $(\alpha4)_2(\beta2)_3$ and 45% $(\alpha4)_3(\beta2)_2$ distribution. This shift from the stoichiometry seen in the absence of nicotine indicates that nicotine drives the assembly of the high sensitivity isoform, $(\alpha4)_2(\beta2)_3$. While some groups have previously hypothesized that nicotine alters the assembly of α4β2 receptors (134,227,228,248), these organelle-specific single molecule studies allowed us to directly observe the process of nicotine altering the assembly of receptors in the ER for the first time. In addition to nicotine altering the assembly of α4β2 within the ER, the percentage of the high sensitivity $(\alpha4)_2(\beta2)_3$ stoichiometry on the plasma membrane was also increased

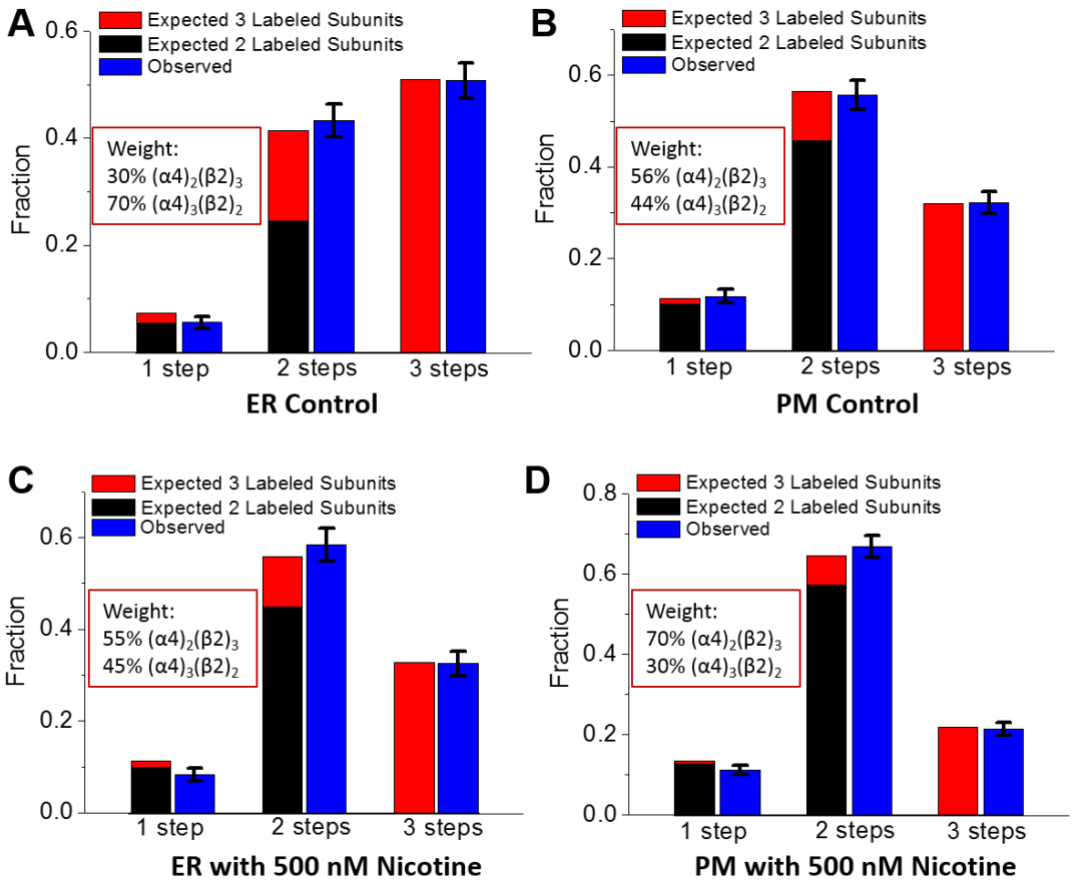

Figure 3.6 Single molecule bleaching step analysis shows organelle-specific differences in α4β2 nAChR isoforms. (A) The observed ratio of vesicles showing one, two, or three steps was 0.057, 0.43, and 0.51, respectively (blue columns). These observed values were then fit to a 30:70 (HS:LS) stoichiometry. The fit was verified using a Chi-square goodness of fit analysis. (B) The expression of α4β2 nAChRs on the plasma membrane fit binomial distributions weighted for 56% $(\alpha4)_2(\beta2)_3$ and 44% $(\alpha4)_3(\beta2)_2$. The observed fraction of vesicles showing one, two, or three bleaching steps was 0.12, 0.56, and 0.32, respectively. (C) For ER resident receptors in the presence of nicotine, the observed fraction of one, two, and three bleaching steps were 0.086, 0.59, and 0.33, respectively. These observed values were then fit to a 55:45 distribution. (D) The observed fraction of vesicles with one, two, or three bleaching steps were 0.11, 0.67, and 0.22, respectively. This was fit to a 70:30 distribution. The error bars for the subunit distribution are based on counting events and are calculated as the square root of the counts.

(Figure 3.6D). After exposure to nicotine, the distribution of photobleaching events obtained from plasma membrane resident α4β2 nicotinic receptors was fit to a distribution of 70% $(\alpha4)_2(\beta2)_3$ and 30% $(\alpha4)_3(\beta2)_2$.

3.3.5 Biased Transfection to Validate Nicotine Induced Shifts in Receptor Stoichiometry

Our studies indicate that nicotine induces a shift toward the assembly of the high sensitivity stoichiometry in both the ER and the plasma membrane. Biased transfection has previously been used to induce a shift in stoichiometry that mimics the effect seen with nicotine (75,249). We performed a set of control studies to verify that in our experiments we could observe a shift in stoichiometry in the ER and the plasma membrane. We transfected HEK293T cells with 1:10 ratio of α4-GFP:β2-wt plasmids and generated vesicles from whole cells, the ER, and plasma membrane. Single molecule photobleaching analysis studies of the unsorted receptors showed 73% $(\alpha4)_2(\beta2)_3$ and 27% $(\alpha4)_3(\beta2)_2$ (Figure 3.7A). The ER originated receptors exhibited 67% $(\alpha4)_2(\beta2)_3$ and 33% $(\alpha4)_3(\beta2)_2$ (Figure 3.7B). The plasma membrane population exhibited 82% $(\alpha4)_2(\beta2)_3$ and 18% $(\alpha4)_3(\beta2)_2$ (Figure 3.7C). These control studies verify that our technique is capable of independently measuring organelle specific shifts in stoichiometry.

3.4 Discussion

We have developed a new technique that allows us to perform organelle-specific single molecule studies on membrane proteins. We utilized this novel method to determine that changes in nicotinic receptor stoichiometry related to nicotine induced upregulation

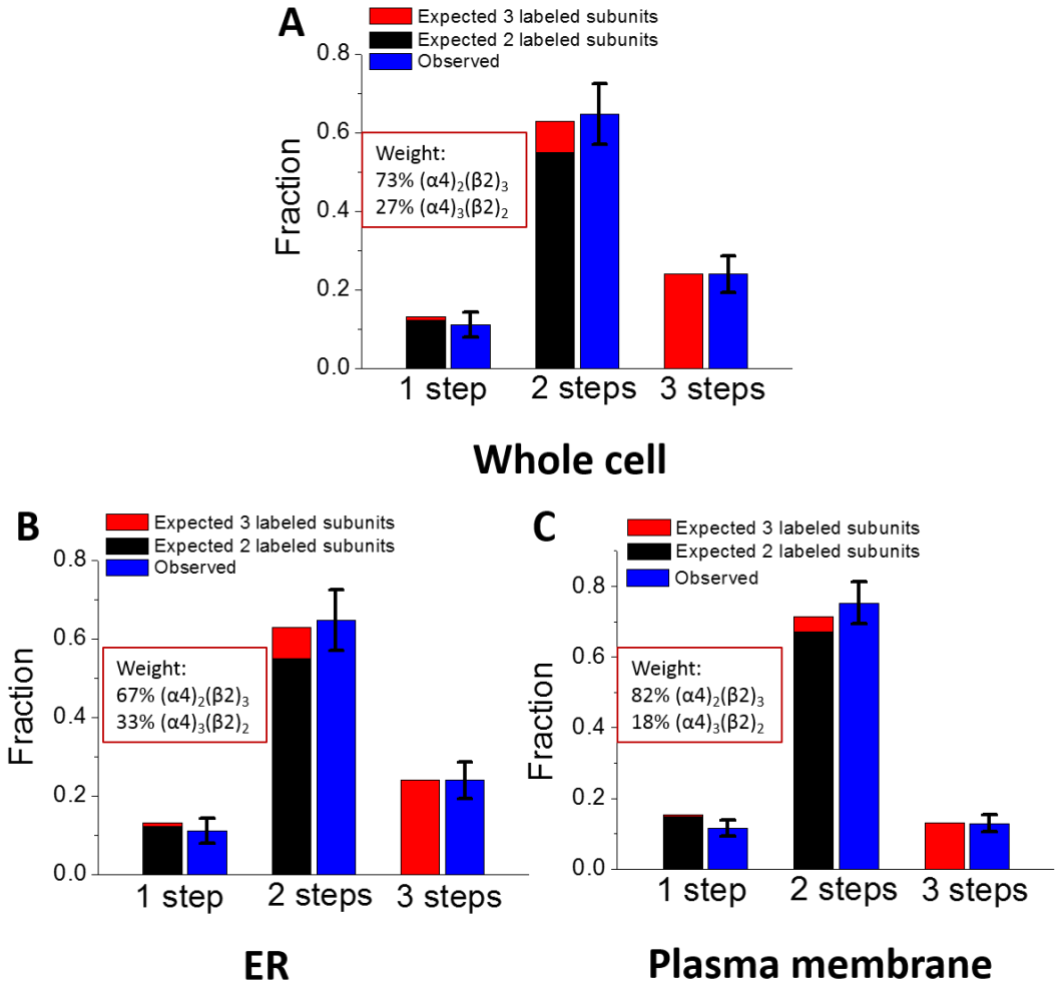

Figure 3.7 Biased transfection of α4β2 to shift assembly towards the high sensitive ($(\alpha4)_2(\beta2)_3$) subtype in both the ER and the plasma membrane. The expected distribution of 1, 2 and 3 photobleaching steps was determined by weighting two binomial distributions and the fit of expected and observed distribution was validated using chi-square goodness of fit test. The assigned weight represent the proportion of the high and low sensitive stoichiometries. (A) The observed photobleaching distribution of the receptors obtained from the whole cell homogenate fit with the expected distribution obtained with 73% $(\alpha4)_2(\beta2)_3$ and 27% $(\alpha4)_3(\beta2)_2$. (B) The ER originated receptors exhibited a photobleaching step distribution which agreed with 67% $(\alpha4)_2(\beta2)_3$ and 33% $(\alpha4)_3(\beta2)_2$ stoichiometries. (C) The stoichiometry for receptors from the plasma membrane was 82% $(\alpha4)_2(\beta2)_3$ and 18% $(\alpha4)_3(\beta2)_2$. The error bars for the subunit distribution are based on counting events and are calculated as the square root of the counts.

(250) are likely driven by both changes in the assembly in the ER and the preferential trafficking of the high sensitivity stoichiometry. The consequences of these changes are an increase in the number of receptors both in the ER and on the plasma membrane as well as a shift in stoichiometric distribution toward the high sensitivity assembly. Previous studies to measure ER specific changes in stoichiometry have primarily been limited by a lack of existing techniques that are capable of directly quantifying subcellular specific structural assemblies of complex proteins in a physiological cellular environment. The isolation of membrane proteins in organelle-specific nanovesicles provides a snapshot of membrane protein assembly in each subcellular location at the time the vesicles are generated. We observed that nicotine, cytisine, varenicline, and bupropion all upregulated the number of receptors on the cell surface. We also observed that these same compounds all altered the stoichiometry of α4β2 nAChRs toward the high sensitivity stoichiometry. Nicotine induced the largest increase in membrane expression and the largest shift toward the high sensitivity stoichiometry, $(\alpha4)_2(\beta2)_3$. The only previous single molecule study of α4β2 stoichiometry on the plasma membrane showed that cytisine elicited a shift toward the low sensitivity stoichiometry in contrast to our findings (10). This previous study only examined the stoichiometry in the very tip of filopodia projected into 150-200 nm apertures. This restricted studies to a specialized surface domain likely accounting for the differences seen here. Our studies sample the entire plasma membrane providing a snap shot of the whole population trafficked to the cell surface. Our results indicate that a wide variety of molecules with different pharmacological properties including agonists, partial agonists, and antagonists all alter receptor assembly. This suggests a possible connection between ligand-induced upregulation and changes in receptor assembly.

We then performed organelle-specific single molecule studies of α4β2 nAChRs in the presence and absence of nicotine to resolve the connection between increased expression and changes in stoichiometry. Comparing the distribution between the low sensitivity stoichiometry and high sensitivity stoichiometry in the ER and plasma membrane showed a much larger fraction of receptors exhibiting the high sensitivity stoichiometry on the plasma membrane. This strongly suggests that the high sensitivity stoichiometry traffics to the cell surface more efficiently than the low sensitivity stoichiometry. We also observed a shift in the ER stoichiometry toward the high sensitivity isoform in the presence of nicotine. This suggests that nicotine induces an intracellular change in the assembly of the α4β2 nAChR. Biased transfection can shift the production of the α4β2 toward the high sensitivity isoform. Single molecule photobleaching event analysis of vesicles from biased transfections experiments confirmed a shift toward the high sensitivity stoichiometry in vesicles originating from the ER. We observed an even larger shift toward the high sensitivity stoichiometry on the plasma membrane. The presence of a higher proportion of the high sensitivity subtype in the plasma membrane compared to the ER verifies the preferential trafficking of the $(\alpha4)_2(\beta2)_3$ stoichiometry from the ER to plasma membrane. Figure 3.8 summarizes the fitted values from Figure 3.6 to illustrate the shift in stoichiometry. The observed differences in organelle stoichiometry show that endogenous assembly in the ER favors the low sensitivity stoichiometry, but that the high sensitivity isoform is preferentially trafficked from the ER to the plasma membrane. Despite having a lower fraction in the ER, this preferential trafficking results in a larger fraction of high sensitivity receptors on the plasma membrane. Recent work by several groups has proposed that nicotine acts as a pharmacological

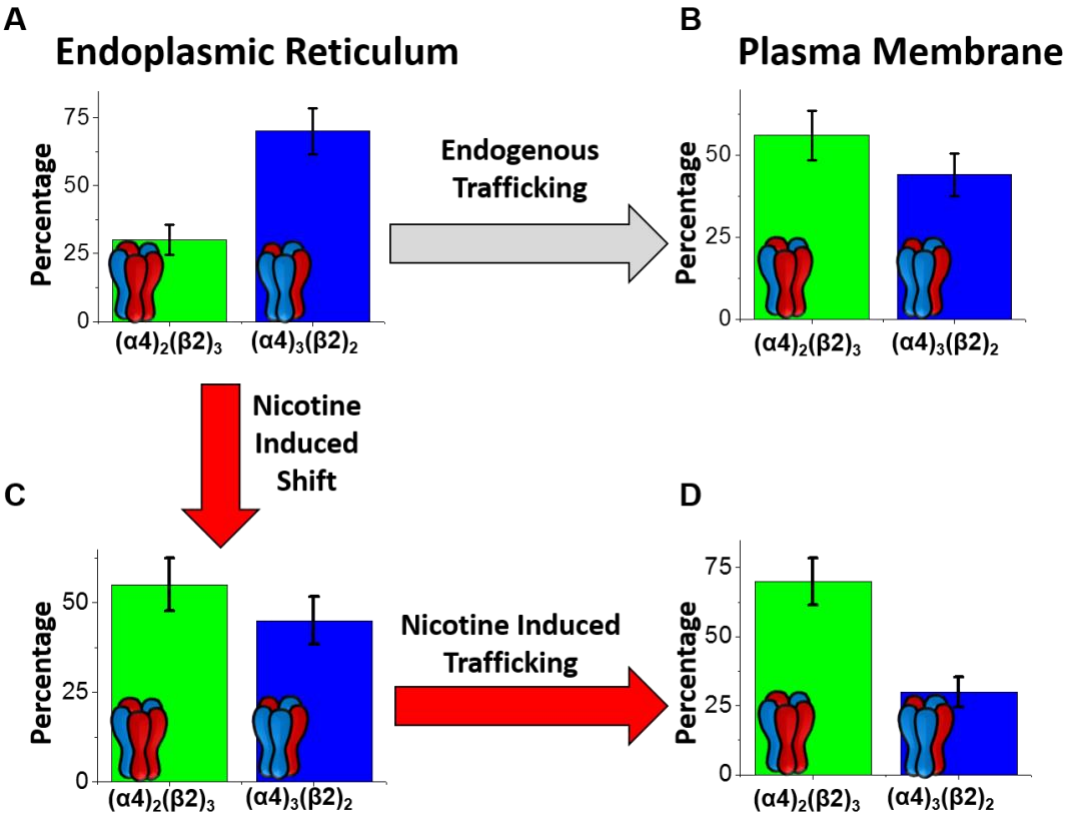

Figure 3.8 Organelle-specific single molecule studies reveal a combination of endogenous preferential trafficking and an intracellular increase in assembly may be responsible for nicotine induced upregulation. (A) Organelle-specific single molecule photobleaching step studies of stoichiometry show that in the absence of nicotine, α4β2 predominately assembles into the 3 alpha stoichiometry (blue) (70%). (B) In the absence of nicotine, the 2 alpha stoichiometry (green) is preferentially trafficked to the cell surface resulting in a higher proportion of receptors on the cell surface having the 2 alpha stoichiometry. (C) In the presence of nicotine, the intracellular assembly of α4β2 is altered to favor the high sensitivity, 2 alpha isoform (green). (D) The increase in availability of the preferentially trafficked stoichiometry, $(\alpha4)_2(\beta2)_3$, leads to an even higher proportion of the 2 alpha stoichiometry (green) on the plasma membrane (70%). The error bars were calculated as the square root of the counts.

chaperone either altering the assembly of nAChRs or influencing the trafficking. Previous studies have also shown that this increases the numbers of α4β2 nAChRs in the ER and the plasma membrane. In these studies, we confirm that increased numbers of receptors in the ER and altered assembly likely play roles in plasma membrane upregulation but are only part of the mechanism. Upon addition of nicotine, the assembly of subunits into a pentamer within the ER is altered to a higher ratio of high sensitivity receptors that can then be efficiently trafficked to the cell surface. This suggests a mechanism of nicotine induced plasma membrane upregulation that is tied to increased numbers of receptors in the ER, preferential trafficking, and a change in assembly. The shift in assembly of α4β2 nAChRs within the ER upon exposure to nicotine towards the preferentially trafficked high sensitivity isoform is likely responsible in part for the nicotine-induced upregulation that has been previously observed. It is possible that residues in the M1-M2 and M3-M4 loops on the intracellular side of each of the subunits of α4β2 regulate preferential trafficking to the cell surface. These intracellular loops contain a number of ER retention and ER exit motifs as well as sites that undergo post translational modification in the secretory pathway (228,251). These same processes are also responsible for targeted trafficking to neuronal subcellular regions. Additionally, recycling from Golgi back to the ER has been shown to be necessary for nicotine induced upregulation of some nicotinic receptor subtypes (140). It is likely that differences in post translational modification sites of the intracellular regions of α4 and β2 lead to the observed differences in trafficking between the two stoichiometries. Employment of our organelle-specific single molecule method enabled the distinction between changes in trafficking compared to changes in assembly of α4β2

nicotinic receptors to partially delineate the underlying mechanism of nicotine-induced upregulation.

Chapter 4

Nanovesicles for Drug Delivery

Contributions: Dr. Rob McCorkle implanted xenografts, injected vesicles and imaged the mice. Hannah Wang and Karli Lipinski assisted to prepare samples for preliminary studies.

4.1 Introduction

Strategies to deliver therapeutics into the human body generally take advantage of the ability of drug molecules to circulate throughout the entire circulatory system. This leads to the interaction of drugs with both healthy and diseased tissue potentially generating undesirable side effects. Researchers have been working to develop a number of artificial drug delivery vehicles that can be utilized to selectively deliver drug molecules to the tissue of interest without affecting off target tissue (143-150). Incorporation of therapeutics into liposomes is the most widely studied approach which has created excitement among researchers after obtaining FDA approval for delivering a chemotherapeutic, doxorubicin, formulated in pegylated liposomes (252). Liposomes are spherical vesicles composed of at least one lipid bilayer and an aqueous center. In this vesicle system, hydrophobic drug molecules can be absorbed into lipid bilayer and hydrophilic molecules can be absorbed into the aqueous center. The lipid bilayer can also be modified with polyethylene glycol molecules to increase therapeutic efficacy of encapsulated drug molecules (252,253). However, these vesicles are rapidly cleared by body's immune system leading to premature degradation of the payload (254). Another lipid based drug delivery system, exosomes, is being extensively studied to utilize advantages of lipid based delivery system while decreasing the limitation of artificial lipid systems such as liposomes (2,255,256). Exosomes are spherical vesicles generated by cells to communicate with other cells and to transfer lipids, proteins, DNA and RNA from donor cells to acceptor cells (1). Exosomes are considered as potential drug delivery vehicles because they are secreted by a wide variety of cells in the human body, retain biomarkers of the cell producing the vesicles, and may selectively carry therapeutics to the cells from which the vesicles were generated (257-

259). Exosomes have potential to be utilized as a personalized treatment system in which patient derived exosomes can be loaded with therapeutics to deliver them to the cells which generated the vesicles (260,261). However, a number of challenges has limited the applicability of exosomes as a drug carrier. One such challenge is to isolate a specific type of exosomes from body fluid which contains hundreds of different types of exosomes generated from the different types of cells in the body. One way to overcome this limitation is to isolate exosomes from cultured cells, but this approach is inefficient because it has been reported that only 0.1 μg of vesicles can be produced from one million cells per day (262). Additionally, this method is costly (39) because cells are usually cultured in a media containing exosome depleted fetal bovine serum (FBS) which is 3-4 times more expensive than normal FBS and a large quantity of cells in a large volume of media usually needs to be cultured to obtain significant quantities of exosomes suitable for drug delivery. Here we present cancer cell-derived vesicles as drug carriers which possess the positive aspects of exosomes including biocompatibility, specificity to targeted tissue and the potential to be utilized in personalized medicine. This new approach can load therapeutics more efficiently, has high yields of vesicle production and can easy be scaled up for commercial production.

To test the feasibility of this strategy, we generated cell-derived vesicles and tested the applicability of these vesicles as general delivery vehicles by separately delivering protein, DNA, and a chemotherapeutic into cultured cells. We then extended these experiments and tested their ability to target implanted A549 cancer cell xenografts in nude mice with compromised immune systems. We performed *in vivo* imaging to locate the position of vesicle delivery. Our data suggested that vesicles could selectively reach the

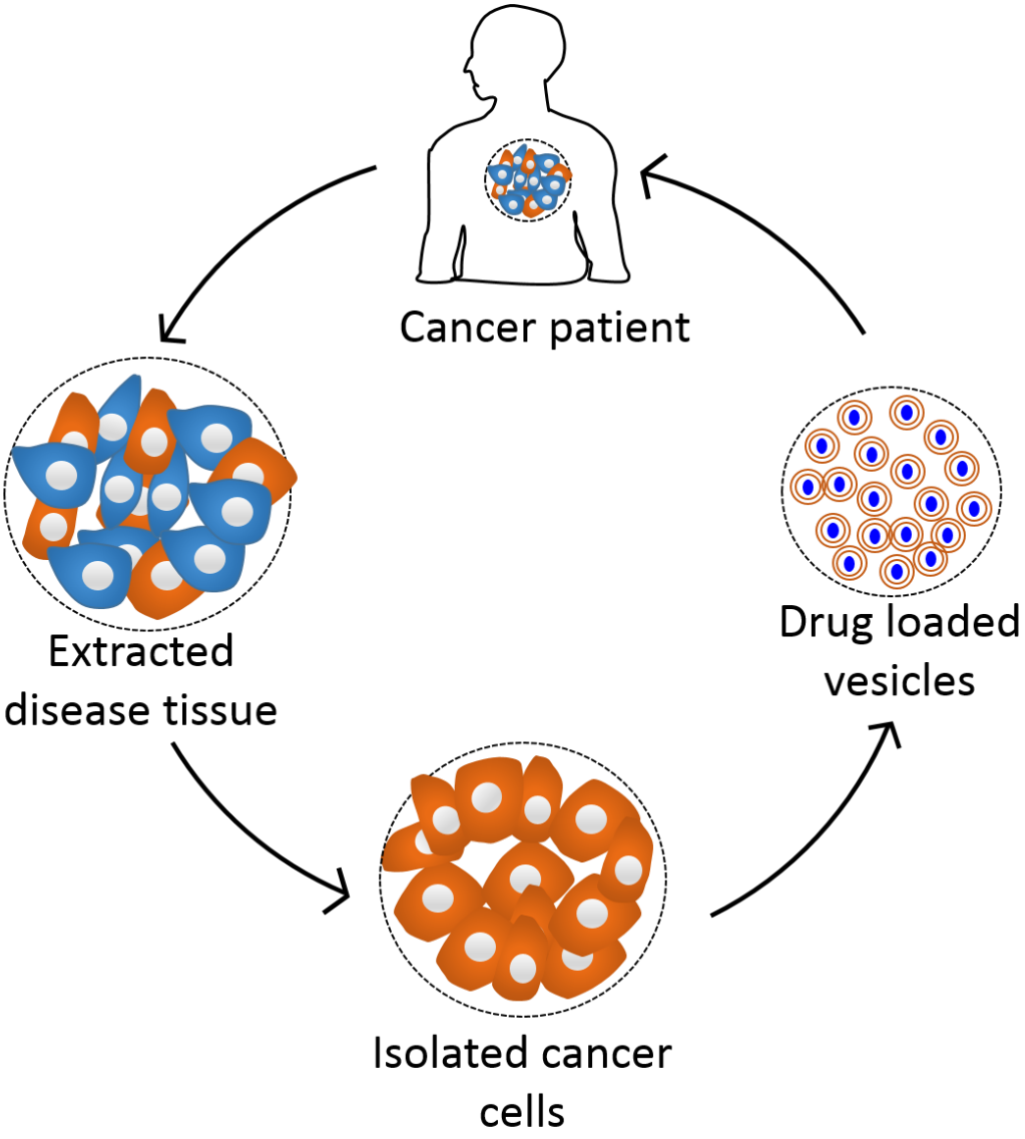

Figure 4.1 An overview of the overall approach of treating cancer patient with cancer-cell derived vesicles of the host. At first, a part of the diseased tissue will be collected through autopsy and cancer cells will be isolated using affinity chromatography. Nanovesicles loaded with chemotherapeutics will be prepared from the cells and will be injected back into the patient. Cancer cells derived vesicles being endogenous to the patient should be able to avoid clearance by the patient's immune system and should selectively deliver the payload to the cancer cells.

xenograft and this finding validates our concept that cell-derived vesicles can be utilized as a drug delivery vehicle. Ultimately, it may be feasible to use this approach for personalized medicine where cancer cells are extracted from a patient and used to prepare vesicles. These vesicles can be loaded with therapeutics and injected back to the patient for targeted delivery (Figure 4.1).

4.2 Experimental Procedures

4.2.1 Preparation of Empty Vesicles

In order to determine the effect of nitrogen cavitation pressure on the size of the vesicles, vesicles were generated with varying nitrogen cavitation pressures of 200, 300, 400, 500, 600, 700, 800, and 900 psi for 5 minutes. Briefly, one flask of HEK293T cells were employed to prepare vesicles for each condition. The cells were detached from the flask by incubating them with 5 ml 0.45 mM EDTA solution in PBS buffer at 37 °C for 5 minutes. The cell slurry was centrifuged at 400×g for 5 minutes and the obtained cell pellet was resuspended in 5 ml of sucrose protease inhibitor buffer (10 mM HEPES, 250 mM Sucrose, pH adjusted to 7.5 and one Pierce protease inhibitor mini tablet per 10 ml buffer). The total number of cells was counted to determine the volume of solution needed to resuspend the vesicle pellet to be generated at the end of the preparation. Vesicles were generated by placing the cells into a nitrogen cavitation chamber (Parr Instruments Company, IL, USA) and applying varying pressure of 200 to 900 psi with 100 psi increment for 5 minutes on ice. The cell lysate was centrifuged at 4000×g for 20 min at 4 °C and the obtained supernatant was centrifuged at 10,000×g for 20 minutes at 4 °C. The resulted supernatant was centrifuged at 100,000×g for 60 min at 4°C to obtain a pellet

containing vesicles. The pellet was then resuspended in PBS buffer so that the vesicles generated from 15 million cells would present into 1 ml PBS buffer.

4.2.2 Extrusion of Vesicles to Reduce Their Size

In order to reduce the size, vesicles were initially generated with 300 psi nitrogen cavitation pressure, unless otherwise mentioned, and the vesicles were extruded with an extruder (Avanti Polar Lipids, Inc., Alabama). Extrusion was conducted for 11 passages each through 200, 100 and 50 nm polycarbonate membrane (Avanti Polar Lipids, Inc., Alabama) in a series. The extruded vesicles were then subjected to dynamic light scattering to determine the size of the vesicles.

4.2.3 Determining the Size of Vesicles with Dynamic Light Scattering

To determine the hydrodynamic diameter of vesicles, the vesicle solution (vesicles generated from 15 million cells were resuspended in one ml PBS buffer) was diluted 1:10 in PBS buffer. About 200 µl of vesicle solution was transferred into a disposable cuvette and the cuvette was placed in a Malvern Zetasizer Nano ZS. Three measurements were taken for 3 minute for each sample. The mean diameter of the vesicles was calculated from the peak value of the fitted curve.

4.2.4 Generation of Protein, DNA and Drug loaded Vesicles

Carboplatin, Dendra2 DNA or proteins (Streptavidin -Alexa Fluor 647) loaded vesicles were generated from HEK293T cells with the same procedure used to prepare empty vesicles, except the sucrose buffer (10 mM HEPES, 250 mM Sucrose, pH adjusted

to 7.5 and one Pierce protease inhibitor mini tablet per 10 ml buffer) used in nitrogen cavitation contained Streptavidin -Alexa Fluor 647 (100 μg/ml), carboplatin (30 mg/ml), or Dendra2 DNA (36.2 μg/ml). Since nitrogen cavitation is believed to generate cell fragments which reorganize to form spherical vesicles, the carboplatin, DNA, proteins or any other materials present in the solution is entrapped inside the vesicles. This is a more efficient approach to loading than either liposomes or exosomes where those vesicles are fully formed and intact prior to drug loading. As a result, those approaches require additional loading steps which are often inefficient and lead to heterogeneity in the concentration of therapeutic in either liposomes or exosomes. The loading process for cell derived vesicles which takes place during formation naturally leads to homogeneous concentrations of encapsulated therapeutics. The free proteins or carboplatin molecules which remained in solution after vesicle formation were removed by passage through a Sephadex G-25 column (PD MidiTrap columns, GE Healthcare). To remove free, proteins molecules, vesicles were resuspended in 5 ml of sucrose buffer and centrifuged at 100,000×g for 1 hour. The supernatant was discarded and the pellet was resuspended in PBS buffer. The volume of the buffer was chosen in way that vesicles generated from 15 million cells were resuspended into 1 ml buffer.

4.2.5 Determination of Carboplatin Concentration in Vesicles

The concentration of carboplatin in the vesicle solution was determined by calculating the amount of platinum present in the solution. At first, 50, 5, 1, 0.5, 0.1, 0.05 ppm platinum standard solutions were prepared in a 5% HCl solution and these solutions were used to obtain atomic emission using inductively couple plasma optical emission

spectrometry (ICP-OES, VISTA-PRO). Carboplatin loaded vesicles were mixed with hydrochloric acid to obtain 5% HCl in the vesicles solution, and this solution was injected with 1 ppm yttrium in a 5% HCl solution. Emissions were recorded at 191.107, 265.945, 204.939, 214.424, 177.648 and 203.646 nm wavelengths to detect platinum and 371.029 nm wavelength to detect yttrium. A standard curve was obtained by plotting the detected emission counts vs the concentration of standard solutions of platinum. The standard curve and emission obtained for a carboplatin solution were utilized to determine the concentration of the carboplatin solution. To validate the accuracy of the concentration of platinum in a sample, an internal standard of 1 ppm Pt solution was added to the sample. When the concentration of platinum in the sample plus the standard was 1 ppm higher than the concentration of Pt in the sample only, the obtained concentration value was considered accurate. The concentration of platinum was converted to the concentration of carboplatin using the molar ratio.

4.2.6 Effect of Carboplatin Loaded Vesicles on the Cell Viability

16 dishes of HEK293T cells (50,000 cells per dish) were plated and cultured in a humidified hood at 37 °C with 5% carbon dioxide. 24 hours later, 2 ml 75 μM carboplatin loaded vesicles were added to 8 dishes and only media was added to the remaining 8 dishes of cells. At 0, 0.5, 1.0, 1.5, 2, 2.5, 3.0 and 3.5-day time points, the total number of live cells present in one dish was counted using a hemocytometer. The live cells were separated from dead cells using a Trypan blue (ThermoFisher, catalog number: 15250061) assay where an equal volume of Trypan blue and cell suspension were mixed. In this assay, the color of

the live cells remained colorless while that of dead cells turned to blue when located using a phase contrast microscope.

4.2.7 Labeling Vesicles with Lipophilic Dyes

The generated empty vesicles were usually mixed with 0.5 to 2 μM lipophilic DiI (ThermoFisher, catalog number: D282) or DiO (ThermoFisher, catalog number: D275) dyes and the mixture was incubated at 37 °C for 30 minutes. The free dye molecules were removed using a size exclusion spin column (PD MidiTrap column, GE Healthcare Life Science, catalog number: 28918008). Briefly, the column was equilibrated with 25 ml PBS buffer and then centrifuged at 1000×g for 2 minutes to remove remaining buffer. Then, 1 ml vesicle solution was added on top of the column and centrifuged at 1000×g for 2 min and the eluate was collected. Unless otherwise mentioned, the separation process was repeated with a second column to ensure removal of trace amount of free dye.

4.2.8 Determining Concentration of Vesicles

Empty vesicles generated from HEK293T cells were first dissolved into PBS buffer in a way that 15 million cell generated vesicles were present in 1 ml of buffer. For this experiment, vesicles were diluted 1:4 in PBS buffer and about 2 ml of empty vesicles were mixed with 0.5 to 1 μM of DiI dye and the mixture was incubated at 37 °C for 30 minutes. One ml of this mixture was passed through a PD MidiTrap column (see section 4.2.7 for details) and the flow through was collected as gel purified vesicle solution. A negative control sample was prepared where DiI was added into 1 ml PBS buffer to make 1 μM DiI

solution and this solution was also incubated at 37 °C for 30 minutes but was not purified with any PD MidiTrap columns.

About 50 µl of this sample was added on top of a coverslip and placed on top of a microscope containing a water immersion objective (LUMPlanFL N, 60X, Olympus). A 532 nm pulse laser source of 20 µW power was directed to the sample through the objective. As fluorescent molecules pass through the confocal beam generated by the objective, the fluorescence intensity fluctuates which was recorded by an avalanche photodiode, τ-SPAD single photon counting module (PiCoQuant GmBh, Germany) and PicoQuant Symphotime 64 (PiCoQuant GmBh, Germany) software. Autocorrelation of the signal was determined to calculate the number of particles present in the focal volume. To determine the concentration of particles, the focal volume needs to be determined as well. Tetraspeck (100 nm, ThermoFisher Scientific, catalog number: T7279) beads of known concentration (1 nM) and diffusion coefficient (0.044×10^{-6} cm^2s^{-1}) were used to determine the focal volume of the confocal bream.

4.2.9 Preparation of Vesicles from RAW Cells

About 65 million RAW 264.7 cells were used to prepare vesicles with 300 psi nitrogen cavitation pressure for 5 min as described in section 4.2.1. The vesicles pellet obtained after 100,000×g centrifugation was resuspended in 2 ml sucrose buffer (details in section 4.2.1). 5 µl 2 mM DiR dye (Thermofisher Scientific, Catalog number: D12731) was added into the 2 ml vesicle solution and the mixture was incubated at 37 ℃ for 30 minute. The unbound dye molecules were removed using an OptiPrep gradient solution

which was prepared by adding 2 ml 50 % OptiPrep and followed by 2 ml 10 % OptiPrep solution into an Ultra-Clear centrifuge tube (Beckman Coulter, catalog number: 340061). Then, the 2 ml vesicle solution with DiR was added on top of the OptiPrep gradient solution. This tube with the gradient solution was centrifuged at 112,000×g for 60 min at 4 °C. A Variable-Flow Peristaltic Pumps (Fisherbrand, ThemoFisher) was used to collect the vesicles solution from the gradient as follows (263). Inlet tubing of the pump was inserted into the Ultra-Clear tube so that the tubing can reach to the bottom-center of the solution without mixing the layers. Then, 1.5 ml solution was pumped out and discarded, and next 1 ml solution was collected. To remove the OptiPrep from the vesicle solution before injecting it into a mice, a PD MidiTrap column was used as described in section 4.2.7.

4.2.10 Mouse Xenograft implantation and Vesicles Injection

In order to develop a xenograft on a mouse, about 200 µl of 12.5 million per ml A459 cells (i.e. 2.5 million total cells) in 50% Matrigel was injected under the skin of an immune deficient nude mouse (Jackson Laboratory, Jax 007850). The xenograft was allowed to grow to become sufficiently large of subsequent experiments. Vesicles were generated from RAW 234.7 cells and labeled with DiR dye as described in section 4.2.9. About 200 µl of the vesicle solution was injected into the tail vein of a mouse containing the xenograft. For positive control experiments, the same volume of vesicle solution labeled with DiR was directly injected into the xenograft of a mouse. For negative control experiments, DiR solution of the similar concentration was passed through a PD MidiTrap column and 200 µl solution was injected though tail vein of a mouse with a xenograft. The

mouse was imaged before injection, immediately following injection, and after 24, 48 or 66 hours of injection using 710 nm excitation and 750 nm emission with an *in vivo* imaging system (IVIS) (Xenogen 50, PerkinElmer, Inc.).

4.3 Results

4.3.1 Vesicle Characterization

We have previously utilized nanovesicles to isolate single receptors in their endogenous membrane (239,263-265). This same technique can be utilized to prepare vesicles loaded with therapeutics to deliver into a targeted location. During vesicle preparation, when cells are ruptured using nitrogen cavitation, cell fragments are generated momentarily and these fragments are spontaneously reorganized to produce vesicles (35,36). Since these vesicles encapsulate the solution where the cells were maintained, we can maintain the cells with a solution containing chemotherapeutics. The vesicles generated through this process would be loaded with the same concentration of therapeutics in the initial solution (Figure 4.2). Since nanoscale particles are absorbed by biological systems faster and more efficiently than microscale particles (266,267), we explored the conditions capable of forming the smallest vesicles. Therefore, I prepared 8 vesicle samples from 8 flasks of HEK293T cells with 200-900 psi in 100 psi increment. The size distribution of the vesicles was determined using dynamic light scattering (DLS) (Figure 4.3) in which sizes of particles moving into a solution are determined based on the motion of the particles. Larger particles move slower while smaller particles move faster and the ability of light scattering by moving particles is dependent on the size of the particle. Hence, in DLS, particle size is determined by monitoring scattered light of a laser source.

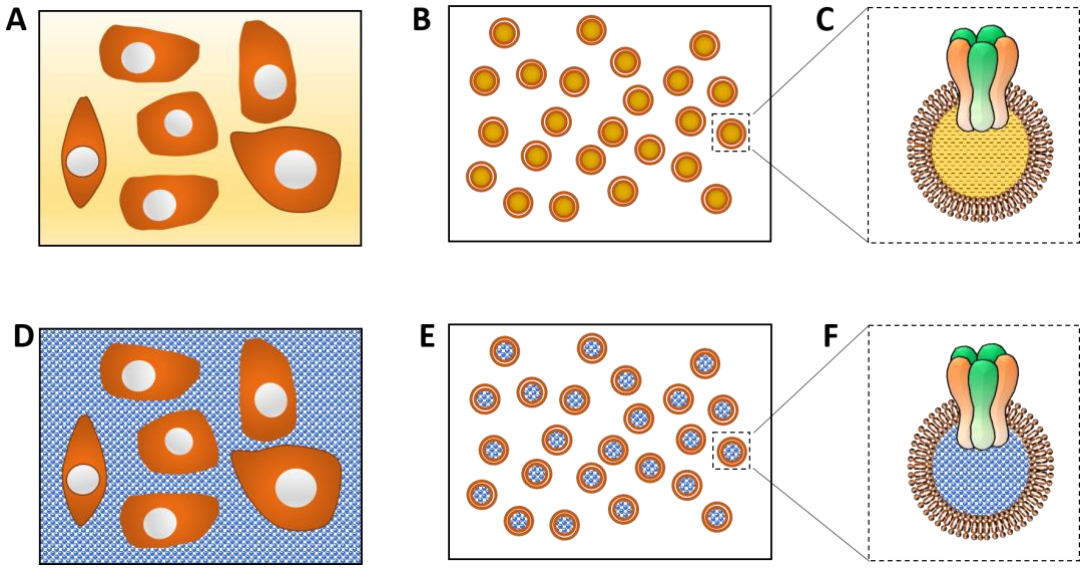

Figure 4.2 A cartoon presenting the evolution of our concept for encapsulating therapeutics into cell-derived vesicles. (A) Previously, we kept cells in a solution (yellow color in A) and upon nitrogen cavitation, we obtained vesicles (B) which encapsulated the solution (yellow center of vesicles, in B). (C) A zoomed-in vesicle would display lipid bilayer holding proteins and encapsulating the cell suspension solution. In the same vesicle preparation technique, if we maintain therapeutics in the cell suspension (blue color in D), upon vesicles preparation, vesicles would encapsulate the therapeutics (E). A zoomed-in image of a vesicle would look similar to the image of C, except this vesicle would entrap therapeutics inside (blue color in F).

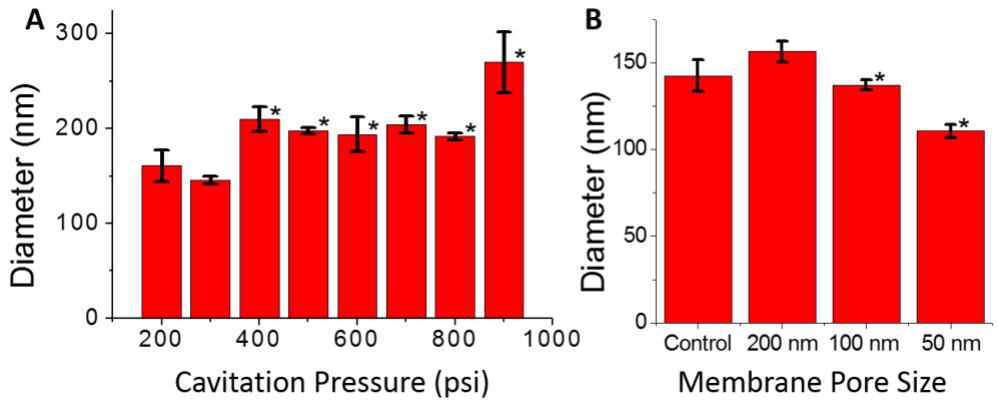

Figure 4.3 Variation of the size of vesicles with the change in nitrogen cavitation pressure and the pore size of membrane used in extrusion. (A) The effect of nitrogen cavitation pressure on the size of the vesicles. 300 psi nitrogen cavitation generated the smallest vesicles and an increase in nitrogen cavitation pressure resulted in an increase in vesicle size. *, $p<0.05$ compared to 300 psi. (B) Determining the effect of extrusion in the size of the vesicles. The vesicles obtained from 300 psi nitrogen cavitation (control in B) were extruded through 200, 100 and 50 nm pore containing polycarbonate membranes. Extrusion with 100 and 50 nm pore containing membrane significantly reduce the size of the vesicle while later one produced smallest vesicles. *, $p<0.05$ compared to control sample. The error bar presented as standard deviation. P-value was calculated using t-test.

The average sizes of the vesicle solution of triplicate measurements were determined (Figure 4.3A). The data showed that 200 psi nitrogen cavitation generated vesicles of 161 nm in diameter while 300 psi produced vesicles of 145 nm. Nitrogen cavitation pressure of 400-800 psi produced vesicles of about 200 nm in size and 900 psi generated the largest vesicles (270 nm in diameter). Ultimately, the data indicated that 300 psi generated the smallest vesicles while any increase in pressure thereafter increases the size of the vesicles.

4.3.2 Effect of Extrusion in the Content of Vesicles

Since nitrogen cavitation generates vesicles larger than 100 nm in size and smaller vesicles are more appropriate as drug delivery vehicles, extrusion can be employed to reduce the size of the vesicles. We have utilized the vesicles obtained from 300 psi nitrogen cavitation pressure and passed those vesicles though a series of polycarbonate membrane filters. Dynamic light scattering was utilized to determine the size of the vesicles. The 200 nm membrane does not change the size of the vesicles since the mean size of the vesicles (150 nm) was smaller than the membrane pore size. 100 nm and 50 nm membranes reduce the size significantly, but the 50 nm membrane generated the smallest size with about 110 nm in diameter (Figure 4.3B). Therefore, an extrusion of vesicles though a 50 nm membrane can be applied to reduce the size of the vesicles.

However, it is necessary to probe if the extruded vesicles can keep the cargo intact. Hence, I prepared vesicles loaded with fluorescein dye by nitrogen cavitation of HEK293T cells at 600 psi in the presence of 1 mM fluorescein dye. A pressure of 600 psi was found to provide large vesicles which can be easily visualized with a total internal reflection

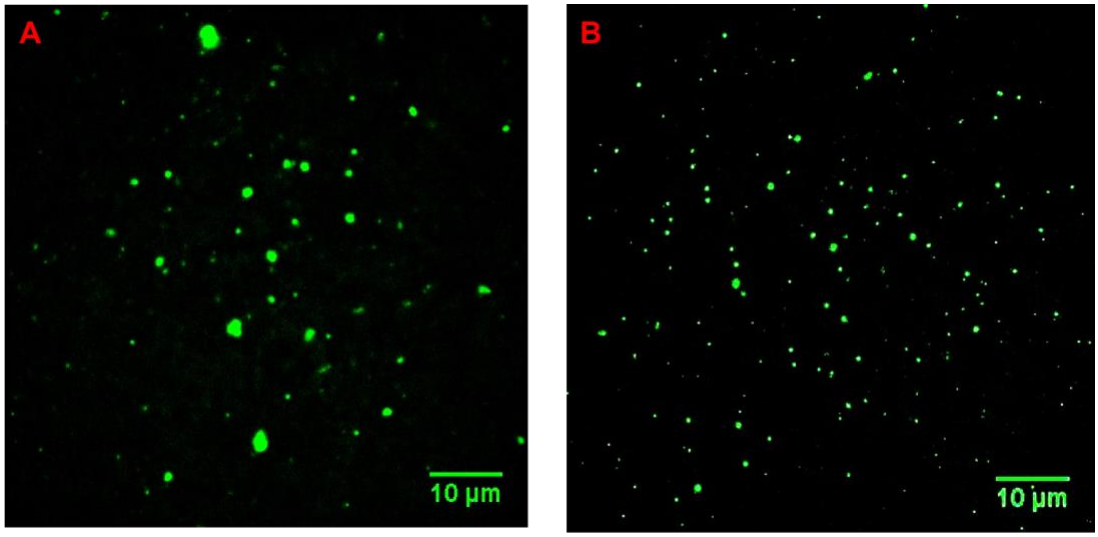

Figure 4.4 Probing whether extrusion of vesicles can maintain the content of the vesicles. (A) An image of vesicles loaded with fluorescein dye molecules. The well distributed large size of the fluorescent spots indicated that fluorescein molecules were present inside the vesicles. (B) An image of fluorescein dye loaded vesicles after extruding through a 100 nm polycarbonate membrane. The relative smaller size of spots indicated the size of the vesicles became smaller and the extrusion did not remove their content.

fluorescence (TIRF) microscope. The fluorescein loaded vesicles were absorbed on a glass substrate and imaged with a TIRF microscope. The presence of bright fluorescent spots on the image clearly indicated the encapsulation of dye molecules inside the vesicles (Figure 4.4). These vesicles were then extruded through a 100 nm polycarbonate membrane, and the extruded vesicles were absorbed on a coverslip and imaged to locate the vesicles. The resultant image indicated that the size of the vesicles was smaller than those of unextruded vesicles, and the extruded vesicles can hold the content inside. Therefore, extrusion can be employed to reduce the size of vesicles without losing cargo.

4.3.3 Effect of Size Exclusion Chromatography in the Concentration of Vesicles

In order to remove free dye or other free cargo molecules from the vesicle preparation, we filtered the vesicle solution through size exclusion columns (PD MidiTrap columns containing Sephadex G-25). Since the vesicles can interact with the beads of the columns, it was necessary to determine if the interaction caused any retention of the vesicles inside the column. Hence, at first, I generated vesicles from HEK293T cells and labeled the lipid bilayer of the vesicles with 0.5 to 1.0 μM DiI dye. The dye labeled vesicles were subjected to fluorescence correlation spectroscopy (FCS) to determine their concentration (Figure 4.5A). In FCS, a confocal beam is directed into a dilute solution containing fluorescent molecules. When a molecule diffuses through the confocal beam it is excited and the resulting fluorescence intensity fluctuation are recorded. The signal is autocorrelated with itself to determine the number of particles present on average in the focal volume. I determined the focal volume using Tetraspeck beads of know concentration, size and diffusion coefficient. Using the focal volume and the number of

molecules in the focal volume, the concentration of the fluorescent particles was calculated. The concentration of vesicles (obtained from 15 million cells and resuspended in 1 ml buffer) was determined to be 150 ± 31 nM. Then, the same sample was passed through a size exclusion column to remove free dye molecules. In this sample, the concentration of vesicles was found to be 157 ± 39 nM. These concentrations were determined not to be statistically significantly different (n=7 for each sample). Since, the first measurement was acquired without removing free dyes from the sample, I recorded the change in fluorescence intensity for a dye solution only and found no autocorrelation in the signal. This result indicated that the signal of the dilute fluorescent dye is not detectable. We observed the same concentration of vesicles before and after size exclusion chromatography indicating this approach can be employed to remove free dyes form the mixture.

4.3.4 Delivering DNA, Protein, and Drug Molecules to Cells

DNA polymers are commonly delivered to cells by entrapping them inside cationic lipid solutions, such as lipofectamine, which can cross the cell membrane. Protein expression generated from the delivered DNA is assessed to validate transfection of the DNA. I wanted to test the applicability of cell-derived vesicles as general delivery vehicles by delivering cargo into cultured cells. I performed a series of experiments to deliver different cargo including DNA, protein, and therapeutics via cell-derived vesicles. First, I prepared vesicles loaded with DNA plasmid for the fluorescent protein Dendra2. This was done using nitrogen cavitation of HEK293T cells in the presence of the DNA containing solution. The cavitation process momentarily generates cell fragments which

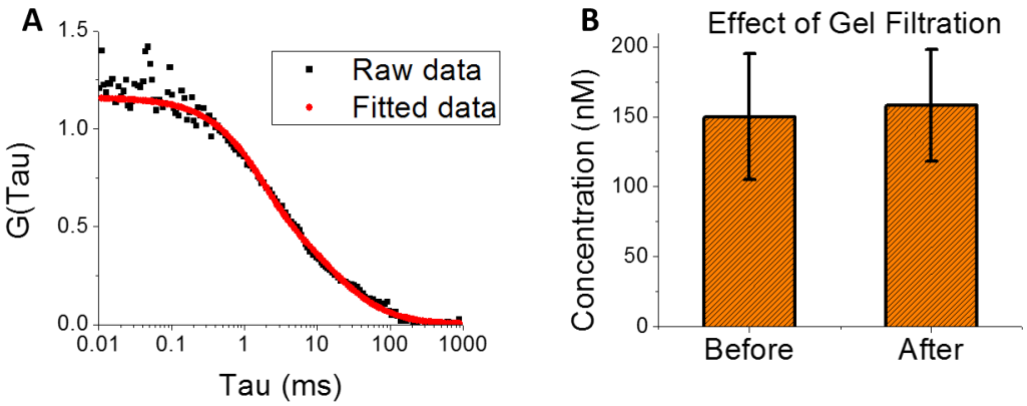

Figure 4.5 Determining the effect of size exclusion chromatography (gel filtration) in the concentration of the vesicles. Vesicles were generated from HEK293T cells with 300 psi nitrogen cavitation pressure and treated with a lipophilic dye (DiI) to label the surface of the vesicles. Fluorescence correlation spectroscopy (FCS) was employed to determine the concentration of vesicles before and after running through gel filtration columns. (A) A representative curve of FCS whose y-intercept value is employed to determine the total number of molecule in the focal volume and the focal volume was determined using standard fluorescence molecules. (B) Concentrations of vesicles before and after passing through a Sephadex G-25 column were determined to be 150 and 160 nM which are not statistically significantly different. Error bars are presented as standard error of the mean.

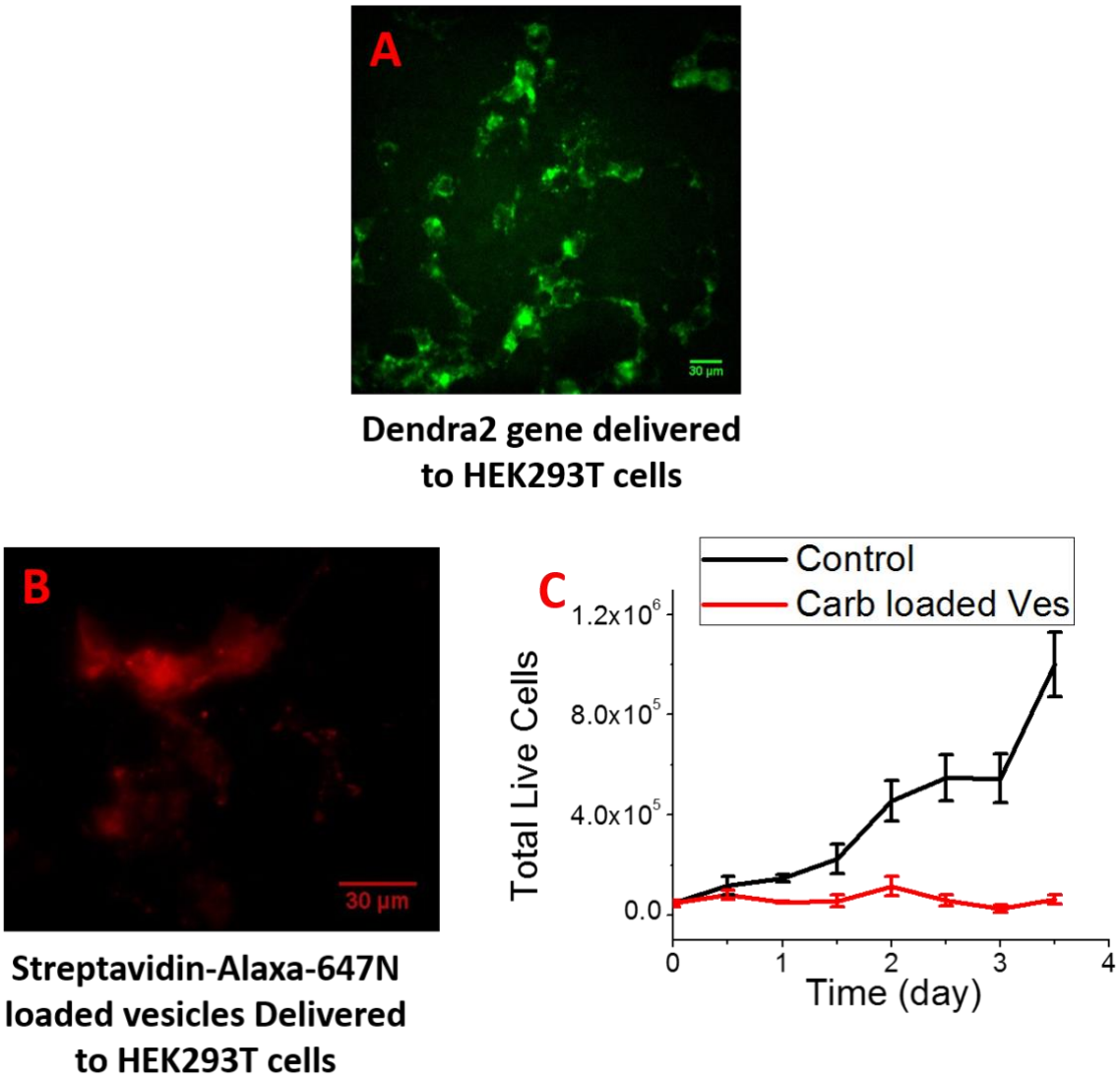

Figure 4.6 Delivery of the DNA of a fluorescent protein (Dendra2), dye conjugated protein, and therapeutic loaded vesicles into HEK293T cells. (A) HEK293T cells were treated with Dendra2 plasmid loaded vesicles for 24 hours, and an image was taken exciting the fluorescent proteins. (B) Alexa Fluor-647 conjugated Streptavidin protein loaded vesicles were added on top of HEK293T cells. The presence of fluorescence signal on the cells indicates the delivery of the protein into the cells. (C) Carboplatin loaded vesicles were added on top of HEK293T cells and the total number of live cells were counted over 3.5-day time periods. Although the total number of live cells for control sample increases exponentially, that of the carboplatin loaded vesicles treated cells remained almost constant. We conclude that the cargoes of the vesicles were delivered into the cells. The error bars are presented as the standard deviation.

spontaneously form vesicles entrapping the cavitation solution. I treated HEK293T cells with the DNA loaded vesicles and simultaneously allowed the cells to grow and express proteins from the delivered DNA for 24 hours. The cells were imaged with a total internal reflection fluorescence microscope. The presence of bright fluorescence in the cells indicated the presence of the Dendra2 fluorescence signal validating the delivery of the DNA into the cells (Figure 4.6A). I then set up a set of experiments to determine if protein could also be delivered into cells. Streptavidin proteins conjugated with Alexa Fluor 647 were similarly loaded into vesicles, and the vesicles were added to HEK293T cells plated on a glass bottom Petri dish. The cells were imaged with a fluorescence microscope exciting the Alexa Fluor 647 dyes to locate the position of delivered proteins. The bright fluorescence in the cells indicated the successful delivery of protein into the cells via cell derived vesicles. Finally, carboplatin loaded vesicles were prepared by nitrogen cavitation in the presence of 30 mg/ml carboplatin in sucrose protease buffer (10 mM HEPES, 250 mM Sucrose, pH 7.5, one Pierce protease inhibitor mini tablet per 10 ml buffer). The concentration of carboplatin in the vesicle solution was determined using an inductively coupled plasma optical emission spectrometry (ICP-OES). 75 μM of carboplatin containing vesicles were added to HEK293T cells and the total number of live cells present on a dish was counted using trypan blue assay. The number of live cells increased dramatically for the control sample (no treatment) whereas that of the carboplatin loaded vesicles treated cells remained almost constant. The difference between the total number of live cells for these two conditions indicates the effect of carboplatin delivered via vesicles. These studies indicate that cell-derived vesicles can be employed to deliver cargo into cells.

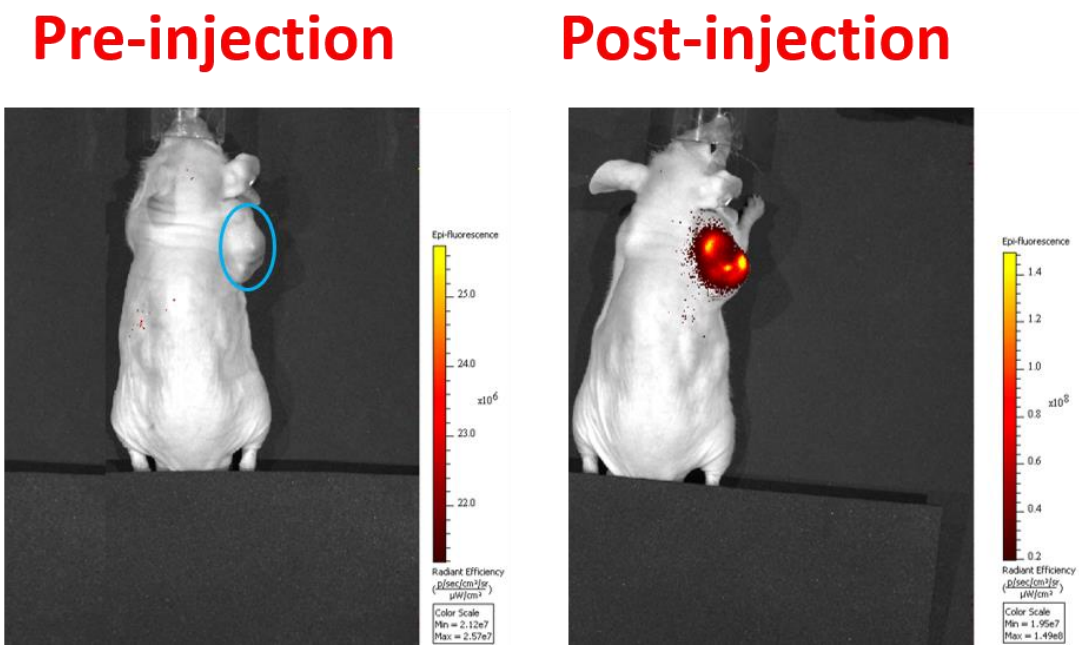

Figure 4.7 Probing that in vivo imaging can be employed to locate fluorescence signal when dye labeled vesicles reach to the tumor. Vesicles were generated from A549 cells and labeled with DiR dye. The vesicle solution was directly injected into the tumor and the mouse was imaged in vivo at pre-injection (left) and 48-hour post injection (right). The presence of fluorescence signal on the tumor up to 48 hours indicates that if intravenous injection of dye labeled vesicles reach the tumor we will be able to locate them using an *in vivo* imaging system (IVIS).

4.3.5 Delivery of Vesicles to Tumors

In order to validate that cancer cell derived vesicles loaded with drug molecules can be delivered to tumors *in vivo*, we first implanted tumors into immune compromised nude mice by subcutaneous injection of A549 cancer cells. About a month later, vesicles were generated from A549 cells using nitrogen cavitation and were labeled with lipophilic dye molecules- DiR. The dye labeled cell-derived vesicles were injected directly into the tumor of a mouse (Figure 4.7). The mouse was imaged *in vivo* to locate the dye in the tumor, and the presence of fluorescence on the tumor was used as a positive control for the experiments. Next, we injected a mouse with the RAW 264.7 cell derived vesicles through its tail vein. In the same manner, another mouse was injected with DiR solution only to serve as a negative control. The control sample did not display any detectable signal on any part of the body of the mouse but the vesicle sample injected into a mouse showed a very bright signal on the location of tumor even after 66 hours of injection (Figure 4.8). These findings validated the concept that cell-derived vesicles generated from nitrogen cavitation can reach tumors and these vesicles can be utilized to deliver chemotherapeutics into a targeted tumor.

4.4 Discussion

At first, we prepared vesicles from HEK293T cells using nitrogen cavitation and characterized the size of vesicles using dynamic light scattering. We have also characterized the effect of nitrogen cavitation pressure on the size of the vesicles. We found that 300 psi nitrogen cavitation pressure generated the smallest vesicles with a diameter of about 150 nm. When the pressure was increased above 300 psi, the size of the vesicles also

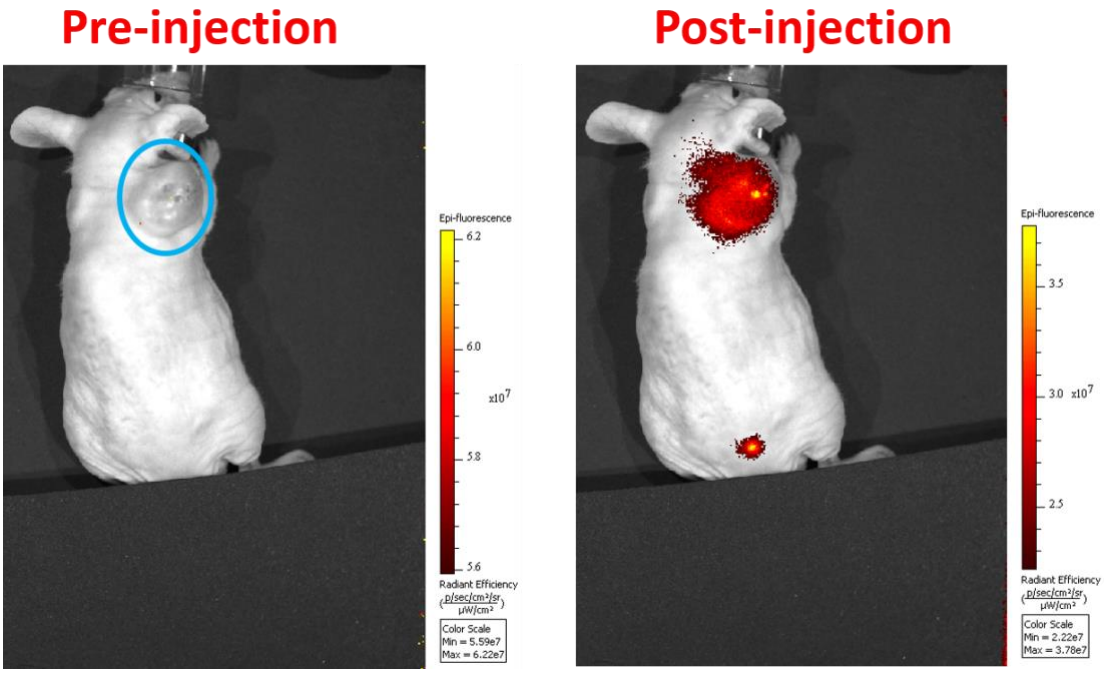

Figure 4.8 Determining whether vesicles can reach into the tumor when the vesicle solution is injected though tail vein of a mouse. Vesicles were generated from RAW 264.7 cells and labeled with DiR molecules. This vesicle solution was injected into a mouse with a xenograft. No signal was obtained in the image captured before injection (circle on the left image indicates the position of the xenograft). After the injection, a bright fluorescent spot was observed in the location of the xenograft (right). This finding validated the concept that cell-derived vesicles using nitrogen cavitation could be used to delivery therapeutics into a targeted location in the body.

increased. We hypothesized that the high pressure causes the cells to rupture unevenly compared to what occurs at 300 psi pressure. Nanoscale vesicles have advantages over larger vesicles in terms of blood circulation time, clearance by the reticulo-endothelial system and delivering of drug molecules through capillaries of the tumors (254,268,269). Hence, obtaining smaller vesicles is more desirable as a drug carrier system (269,270). One of the prominent methods to reduce the size of the vesicles is extruding them through a polycarbonate membrane (38). We employed extrusion with varying pore sizes of polycarbonate membranes to monitor the size change of the vesicles. We demonstrated that 50 nm membranes can produce vesicles of about 100 nm in size which is significantly lower in size than those generated at 300 psi nitrogen cavitation. Other types of vesicles with diameters of 100 nm were reported not to be recognized by the reticulo-endothelial system during *in vivo* delivery of cargo (268,271).

Next, we prepared vesicles from HEK293T cells using nitrogen cavitation and labeled the vesicles with DiI dye molecules which are lipophilic and weakly fluorescent when not bound with lipids. Then, we passed the vesicle solution through PD MidiTrap columns to remove the free dye molecules. We determined the concentration of vesicles using fluorescence correlation spectroscopy and found that the concentration of the vesicles before and after passing through gel columns was around 150 nM. This result suggested that no detectable quantity of vesicles was retained inside a size exclusion column which made them suitable for removing unbound dye molecules from the vesicles solution.

Nitrogen cavitation produces cell fragments which spontaneously reorganize to generate vesicles encapsulating the solution containing cargo molecules. Thus, we made

vesicles loaded with DNA, proteins or therapeutics by maintaining these cargo molecules in the nitrogen cavitation solution. Before testing the delivery of the cargo molecules into the tumor of a mouse model, we tested the delivery to HEK293T cells. The vesicles were added on top of the cells for a given time period and images of the cells were analyzed to determine the delivery of the cargo. For example, Dendra2 plasmid was loaded into vesicles and the vesicle solution was added on top of HEK293T cells. The cells were allowed to grow for 24 hours and then imaged with fluorescence microscopy to locate the presence of Dendra2 fluorescent proteins. Similarly, we demonstrated that vesicles can be employed to deliver proteins and therapeutics to cells. Since cell-derived vesicles are structurally similar to exosomes (40), the mechanism of delivery of cargo by cell-derived vesicles might be similar to that of exosomes. Although, the mechanism of exosome internalization is not clearly understood, it has been reported that exosomes might deliver its cargo into the recipient cell by endocytosis or direct fusion of vesicles into the recipient cells (272).

Next, we validated the concept that therapeutic loaded cell-derived vesicles prepared with nitrogen cavitation can reach a tumor engrafted in an animal. First, we implanted a xenograft into an immune compromised nude mouse with A549 cancer cells. When the tumor grown enough, we generated vesicles from cells and ran a positive control experiment. In this experiment, the cell-derived vesicles labeled with DiR dye molecules were directly injected into the tumor. The mouse was imaged with an *in vivo* imaging system, and we compared the images obtained from subsequent imaging session to probe the delivery of vesicles into the tumor. Then, we injected DiR dye molecules containing solution into the tail vein of another mouse with a xenograft and found no detectable

location of dyes in the tumor. Finally, we prepared vesicles labeled with DiR molecules and injected this sample into the tail vein of a xenograft containing mouse. Images of the mouse showed that vesicles can successfully reach to the tumor. The result validated our hypothesis that cell derived vesicles obtained from nitrogen cavitation could be utilized to delivery therapeutics selectively to tumors.

4.5 Conclusion

We have demonstrated that nanoscale cell derived vesicles can be generated from mammalian cells with a combination of nitrogen cavitation and extrusion. We have employed these vesicles to deliver plasmids, chemotherapeutics, and proteins into cells. Then we made cell derived vesicles labeled with an organic dye and injected these dye labeled vesicles into mice engrafted with a tumor. Our preliminary data suggests that the vesicles can successfully recognize and reach cancer cells. In the future, we can extend this study to patient derived xenografts. In this new study, we plan to isolate cancer cells from a cancer patient through biopsy, and a fraction of the cells will be implanted into a mouse to develop a xenograft. The remaining cells will be cultured and used to prepare vesicles which will be labeled with dye molecules and injected into the mouse to check if the vesicles can reach the xenograft. Chemotherapeutic loaded vesicles will also be injected into the mouse to determine the effectiveness of the drug vs those injected without any delivery systems.

Chapter 5
Conclusions and Future Directions

5.1 Conclusions

One of the main goals of my graduate research was to develop a method to isolate single transmembrane proteins into a diffraction-limited spot without compromising its functional and structural integrity. Single molecule fluorescence methods can provide detailed knowledge about the conformational changes of biomolecules (4,5), dynamics of the gating of ion channels (6), folding and unfolding of proteins (7,11) and assembly of oligomeric receptors (10,13). In order to obtain these details of molecular events, single transmembrane proteins need to be isolated into diffraction limited spots. One common approach to isolating single transmembrane proteins involves expressing a very low level of proteins in mammalian cells or oocytes. The low expression rate can reduce the probability of isolating multiple proteins on a diffraction limited spot (13). However, the cells tend to produce auto-fluorescence which increases the background fluorescence level and thus reduces signal to noise ratio (15,16). Additionally, membrane proteins tend to move laterally along the membrane increasing the local concentration of the protein (273). This phenomenon along with low signal to noise ratio difficult the single-molecule studies of transmembrane proteins located in live cells. Another common approach involves isolation of the membrane protein into an artificial lipid bilayer (273). One of the intermediate steps of this approach involves isolation of the protein into a detergent solution which endangers the structural integrity of oligomeric transmembrane proteins. To circumvent the issue associated with the concentration barrier, the low signal to noise ratio, and the structural integrity, I have expressed transmembrane proteins in mammalian

cells with a very low expression level and then isolated the proteins into cell-derived vesicles without removing them from their physiological membrane. In Chapter 2, I utilized cell-derived vesicles prepared with nitrogen cavitation to isolate single membrane proteins. I showed that cell-derived vesicles can be employed to study stoichiometric assembly of CFTR, EGFR, and nicotinic receptors. We determined the stoichiometry of α3β4 nicotinic receptor with single color and two colors experiments. We also demonstrated that receptors isolated in cell derived vesicles are functional. Therefore, proteins isolated in cell-derived vesicles can maintain their functional and structural integrity.

Although single molecule studies of transmembrane proteins can provide an overview of the structural properties of a protein, organelle specific information is sparse due to the lack of a proper methods of isolating membrane proteins from different organelles. The ER is the organelle where transmembrane proteins become synthesized and assembled and the assembled proteins are transported from the ER to the plasma membrane. Isolation of oligomeric receptors into cell-derived vesicles of different organelles can provide information of the freshly assembled receptors and the already trafficked receptors. When a small molecule interacts with nascent receptors to affect their structural and functional properties, the organelle specific vesicles, in combination with single-molecule receptors, can provide detailed information about the drug induced intracellular changes in the assembly of the receptor. In Chapter 3, in collaboration with another member of Richards' lab (Dr. Ashley M. Loe), I studied the effects of different types of nicotinic receptors' ligands on the assembly and trafficking of α4β2 nicotinic receptor. We demonstrated that all classes of ligands – agonist, partial agonist, and

antagonist – can increase the total number of receptors in both the ER and the plasma membrane and can alter the distribution of the receptors in the ER and the plasma membrane. The ER and plasma membrane-originated receptors were isolated to study the stoichiometric assembly in those organelles. The data suggested that the ER has a higher proportion of the low sensitivity isoform of α4β2 nicotinic receptors than the plasma membrane. The opposite is true for high sensitivity isoforms. This result suggested that the high sensitivity isoform trafficked more efficiently from the ER to the plasma membrane than their counterparts. When nicotine was present, the ratio of a high sensitivity isoform increases in both the ER and plasma membrane. This result suggested that nicotine helped the high sensitivity isoform to be assembled more efficiently in the ER and the same isoform, which was higher in number in the pool due to the presence of nicotine, was preferentially trafficked to the plasma membrane, increasing the proportion of the isoform than that of the no nicotine treatment.

The cell-derived vesicles employed in this study are structurally similar to the lipid-based drug carriers used to deliver drug molecules into a targeted location. Some researchers have employed cell-derived vesicles to deliver therapeutics into tumors of mice. However, our method of vesicle preparation is faster and can be utilized to load with drug molecules more efficiently. Hence, my final aim was to explore the applicability of cell-derived vesicles as a drug delivery system. An ideal drug carrier should possess the ability to avoid seizure and degradation by body's immune system, should reach to the targeted tissue of the body and should penetrate the cellular system to deliver the cargo for an extended time period (39). A number of approaches has been made to develop an ideal drug delivery system including liposomes, polymer nanoparticles, micelles, carbon

nanotubes and gold nanotubes (143-150). Among all these approaches, liposomes have brought the most excitement by obtaining FDA approval for delivering chemotherapeutics (252). Liposomes are spherical vesicles consisting of at least one lipid bilayer containing a hydrophilic aqueous center. Hydrophilic drug molecules can be loaded into the aqueous center and lipophilic drug molecules can be inserted into the lipid bilayer. Thus, liposomes can be employed to deliver toxic drug molecule into a targeted location. However, liposome can elicit an immune response leading to premature degradation of the payload (274). These vesicles are cleared by the cellular defense system and become degraded prematurely. Hence, another type of vesicle found in the body fluid called exosomes are being extensively studied to employ as a drug delivery system (2,255,256). Exosomes are endogenous vesicles utilized by cells to communicate with cells located at a distance. Since cells spontaneously produce exosomes, the collection of these vesicles is a very time consuming, inefficient and expensive process (39).

In Chapter 4, first, we demonstrated that cell derived vesicles can be employed to deliver genes, proteins and chemotherapeutics into cells by loading these materials inside the vesicles. We implanted xenografts in immune compromised nude mice with A459 cancer cells and prepared vesicles from the cancer cells as well as RAW 264.7 cells using nitrogen cavitation. We labeled the vesicles with a lipophilic dye suitable for *in vivo* imaging. The dye-labeled vesicles were injected into mice with xenografts, and the mice were imaged with an IVIS to locate the position of the dye in the body of the mice. The captured image displayed the presence of a large fluorescent spot on the location of the xenograft indicating successful delivery of vesicles. This finding validated our hypothesis that cell-derived vesicles generated by nitrogen cavitation could be utilized to deliver

therapeutics into a targeted location. We believe that this study can be extended to human subjects where cancer cells can be extracted from a patient and these cancer cells can be utilized to prepare vesicles loaded with chemotherapeutics. These vesicles will be injected back to the patient to deliver chemotherapeutics selectively to the patient's cancer cells.

5.2 Future Directions

In chapter 2, we demonstrated that different types of transmembrane proteins can be isolated on cell-derived vesicles to conduct single molecule studies. We believe that these vesicles can be utilized to isolate all sorts of membrane proteins at the single molecule level, and thus single molecule studies can be extended to transmembrane proteins which otherwise could not be studied using traditional approaches. Although our demonstration of the single molecule studies were limited to step-wise photobleaching and fluorescence correlation spectroscopy, we believe that our approach can be adopted to conduct studies with single molecule FRET of freely diffusing molecules or immobilized molecules. We also believe that single channel kinetics of different ion channels can be studied by isolating them into cell-derived vesicles.

In chapter 3, we revealed that nicotine and other nicotinic ligands can alter the assembly and trafficking of α4β2 nicotinic receptors, it would interesting to know if the endogenous nicotinic receptor ligand, acetylcholine, can also alter assembly and trafficking of nicotinic receptors. I hypothesis that the endogenous ligand will also alter the assembly and trafficking of nicotinic receptor to a certain extend. This experiment will help us to understand if the mechanism of nicotine induced upregulation is any different than that

observed for endogenous ligands. Additionally, I believe the single molecule studies can be extended to understand the effect of nicotine and other nicotinic receptor ligands on the assembly of α4β2 nicotinic receptors derived from brain samples of animal model. In this work, a transgenic mouse model can be developed to knock-in α4-GFP gene and the brain from the mouse can be utilized to prepare vesicles isolating single receptors. Similarly, studies can be conducted with β2-GFP knock-in mouse brains to complement the studies with α4-GFP knock in mouse brain. We can also isolate receptors from presynaptic and postsynaptic neurons and study the assembly of different nicotinic receptors in those regions with and without presence of nicotinic receptor ligands.

In chapter 4, we showed that cell-derived vesicles can be utilized to encapsulate therapeutics and deliver the therapeutics into a tumor selectively. Still, we need to conduct a number of experiments to understand the mechanism of delivery and stability of the vesicles. We can prepare vesicles and store into -80 °C for a given length of time, then the vesicles can be lyophilized to determine the change in stability of the vesicles over the time period using a differential scanning calorimetry. We can also determine change in concentration of vesicles and surface charge or zeta potential over time using FCS and DLS respectively to study the effect of storage in those parameters.

Appendix

Outline of GUI: getTimeTraces_byFaruk.fig

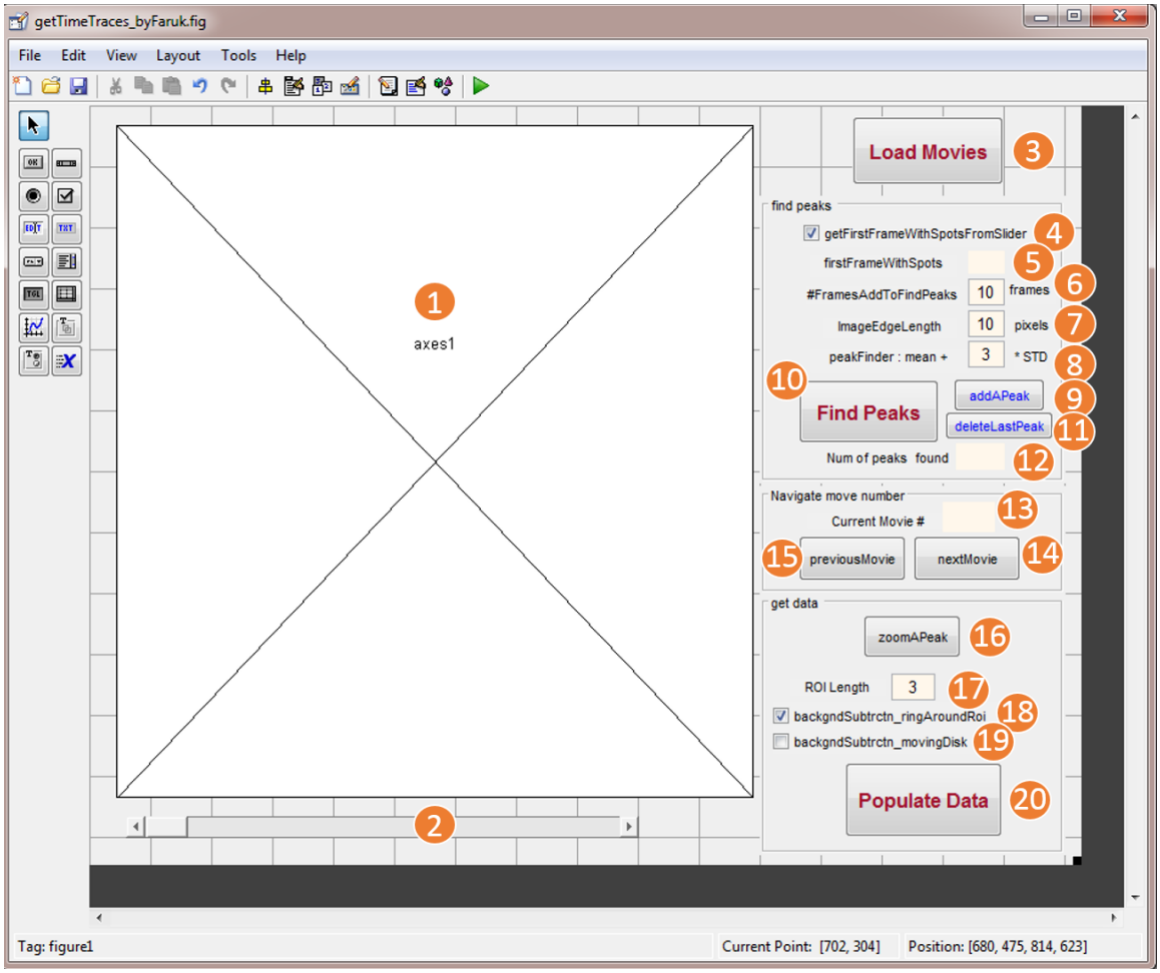

Figure S1: Outline of the GUI used to collect time traces from tiff stacks. This GUI was saved as "getTimeTraces_byFaruk.fig". The underlying variables (Tag) for components 1-20 can be found in the Table S1.

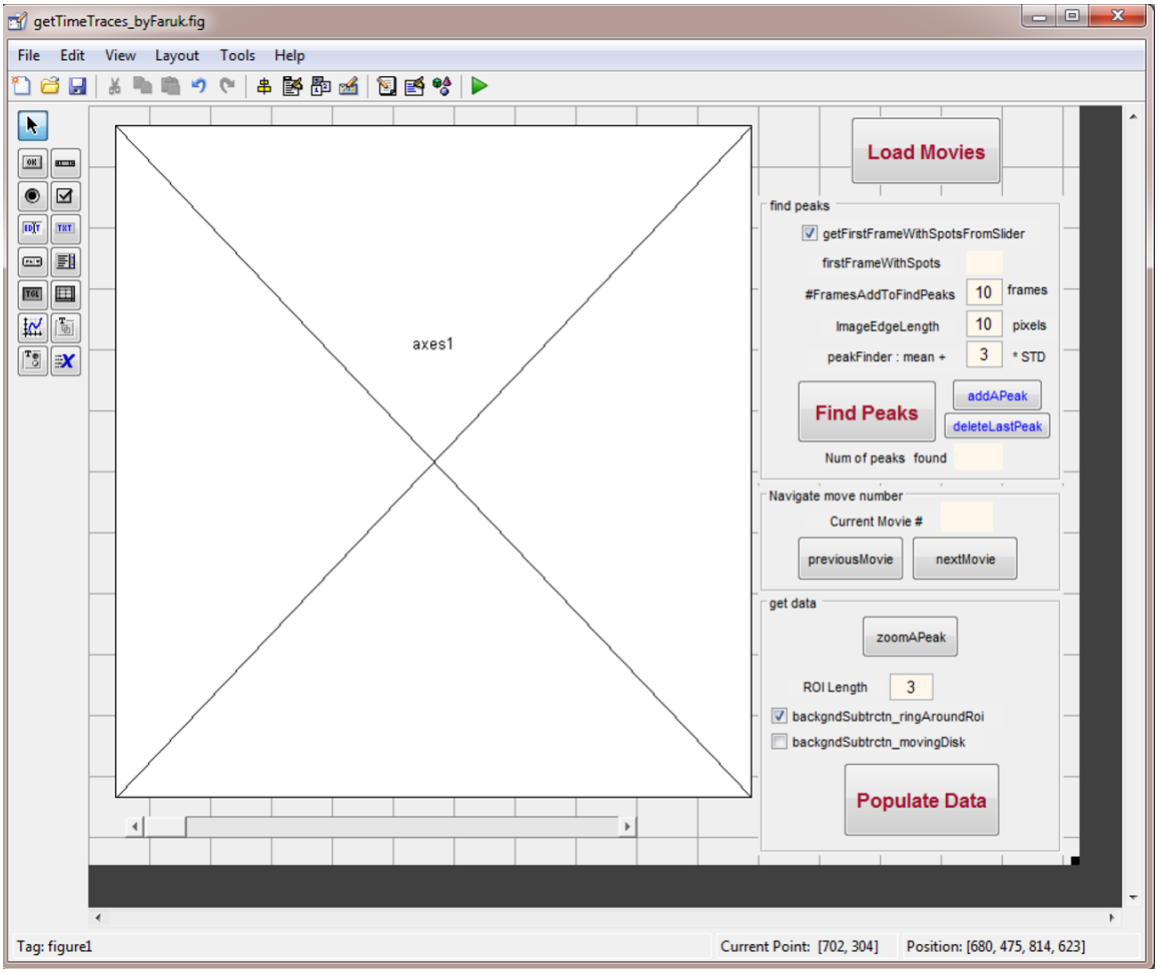

Figure S2: A figure of the GUI shown in Figure S1 without numbering. This figure displays all texts and default values on the GUI.

Table S1: The essential variables used in the figure for GUI (Figure S1). All other components were either Static Texts or Panels.

Number in Figure S1	Type of component	String	Tag
1	Axes	Not applicable	axes1
2	Slider	Not applicable	sliderChangeFrameInPlot
3	Push Button	Load Movies	loadMovies
4	Check Box	getFirstFrameWithSpotsFromSlider	getFirstFrameWithSpotsFromSlider
5	Static Text	Empty	firstFrameWithSpots
6	Edit Text	10	NumOfFramesAddToFindPeaks
7	Edit Text	10	ImageEdgeSize
8	Edit Text	3	stdInput
9	Push Button	addAPeak	addAPeak
10	Push Button	Find Peaks	findPeaks
11	Push Button	deleteLastPeak	deleteLastPeak
12	Static Text	Empty	numOfSpotsFound
13	Static Text	Empty	currentMovieNumDisp
14	Push Button	nextMovie	nextMovie
15	Push Button	previousMovie	previousMovie
16	Push Button	zoomAPeak	zoomAPeak
17	Edit Text	ROI Length	roiLength
18	Check Box	backgndSubtrctn_ringAroundRoi	backgndSubtrctn_ringAroundRoi
19	Check Box	backgndSubtrctn_movingDisk	backgndSubtrctn_movingDisk
20	Push Button	Populate Data	populateData

Matlab Code in a file called, "getTimeTraces_byFaruk.m"

All Matlab codes were written in Matlab (version 2017a). The following code was saved on a Matlab file, called, "getTimeTraces_byFaruk.m".

```
function varargout = getTimeTraces_byFaruk(varargin)
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @getTimeTraces_byFaruk_OpeningFcn, ...
                   'gui_OutputFcn',  @getTimeTraces_byFaruk_OutputFcn, ...
                   'gui_LayoutFcn',  [], ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
  gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before getTimeTraces_byFaruk is made visible.
function getTimeTraces_byFaruk_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   unrecognized PropertyName/PropertyValue pairs from the
%            command line (see VARARGIN)

% Choose default command line output for getTimeTraces_byFaruk
handles.output = hObject;
set(handles.axes1,'XTickLabel','','YTickLabel','','xtick',[],'ytick',[]);
set(handles.figure1,'Name',mfilename); %set mfilename as GUI name
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes getTimeTraces_byFaruk wait for user response (see UIRESUME)
% uiwait(handles.figure1);
```

```
% --- Outputs from this function are returned to the command line.
function varargout = getTimeTraces_byFaruk_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function loadMovies_Callback(hObject, eventdata, handles)
% hObject    handle to loadMovies (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
   pathMfile = fileparts(mfilename('fullpath'));
   addpath(pathMfile);
%     addpath(fullfile(pathMfile, 'timeTrace_calls'));
   %Open standard dialog box for retrieving files
   [fileName,filepath]=uigetfile('*.tif','select the Tiff Movie','MultiSelect','on');

    if iscell(fileName) %multiple files were selected
       cd(filepath);
       initalNumberOfFramesLoaded=getInitalNumOfFramesToLoad();
       totalNumMovies=length(fileName);
       tempRawData=cell(1,totalNumMovies);
       fullFileName=cell(1,totalNumMovies);
       hM=waitbar(0,'Reading Movie : 1');
       for i=1:totalNumMovies
          waitbar(i/totalNumMovies,hM,['Reading                    Movie                    :
',num2str(i),'/',num2str(totalNumMovies)]);
          fullFileName{i}=fullfile(filepath,fileName{i});

tempRawData{i}=tempLoadAMovie(fullFileName{i},i,initalNumberOfFramesLoaded);
       end
    elseif  fileName~=0 %only one file has been selected

        cd(filepath);
        initalNumberOfFramesLoaded=getInitalNumOfFramesToLoad();
        totalNumMovies=1;
        tempRawData=cell(1,1);
        fullFileName={fullfile(filepath,fileName)};
        hM=waitbar(0,'Reading Movie : 1');
```

```
tempRawData{1}=tempLoadAMovie(fullFileName{1},1,initalNumberOfFramesLoaded)
;

    elseif fileName==0; %user did not select any movie
        return
    end
%initiate donor and acceptor peak sets
   handles.peakSets=cell(1,totalNumMovies);
   handles.roiDiaMats=cell(1,totalNumMovies);
   handles.combIm=cell(1,totalNumMovies);
   handles.firstFrameWithSpotsMat=cell(1,totalNumMovies);
   handles.thresholdValueMat=cell(1,totalNumMovies);
   handles.stdInputMat=cell(1,totalNumMovies);
   handles.NumOfPeaksManuallySelected=zeros(totalNumMovies,1);
%share data
   handles.tempRawData=tempRawData;
   handles.fullFileName=fullFileName;
   handles.currentMovieNum=1; %count number of movie under analysis
   handles.totalNumMovies=totalNumMovies;
%plot data
   plotImage(handles.axes1,tempRawData{1}(:,:,1));
%update
   sliderMin=1;
   sliderMax=size(tempRawData{1},3);
   stepSize=[1,1]/(sliderMax-sliderMin);
   set(handles.sliderChangeFrameInPlot,'Min',sliderMin,'Max',sliderMax,'Value',5,
'SliderStep', stepSize);

currMovDis=[num2str(handles.currentMovieNum),'/',num2str(handles.totalNumMovies)]
;
   set(handles.currentMovieNumDisp,'String',currMovDis);

 %if user is analyzing 2nd movie, need to reset the getFirstFrameWithSpotsFromSlider
   set(handles.getFirstFrameWithSpotsFromSlider,'Value',1);
   set(handles.firstFrameWithSpots,'String','1');

close(hM);
guidata(hObject, handles);

function val=getInitalNumOfFramesToLoad()
   prompt = {'Number of frames will be initially loaded ?'};
   dlg_title = 'Input';
   num_lines = 1;
   defaultans = {'50'};
   val = inputdlg(prompt,dlg_title,num_lines,defaultans);
```

```
   val=round(str2double(val{:}));

function
[tempRawData]=tempLoadAMovie(fullFileName,movieNum,initalNumberOfFramesLoa
ded)
      genMessage=['Reading first ', num2str(initalNumberOfFramesLoaded), ' frames of
Movie # '];
      hTemp=waitbar(0,[genMessage,num2str(movieNum)]);
      MovieInfo = imfinfo(fullFileName);
      width=MovieInfo.Width;
      height=MovieInfo.Height;
      numberOfImages = length(MovieInfo);
      if numberOfImages<initalNumberOfFramesLoaded %load only
initalNumberOfFramesLoaded frames
         msgbox('There should be at least', num2str(initalNumberOfFramesLoaded), '
frames per movie','Error','error');
      end
      tempRawData = zeros(height,width,initalNumberOfFramesLoaded,'uint16');
      for k = 1:50
         updatWaitbar=[{['Reading first
',genMessage,num2str(movieNum)]};{[num2str(k),'/',num2str(initalNumberOfFramesLo
aded)]}];
         waitbar(k/initalNumberOfFramesLoaded,hTemp,updatWaitbar);
         tempRawData(:,:,k)=imread(fullFileName, k, 'Info', MovieInfo);
      end
      close(hTemp);
% --- Executes on slider movement.
function sliderChangeFrameInPlot_Callback(hObject, eventdata, handles)
% hObject    handle to sliderChangeFrameInPlot (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
   val=round(get(hObject,'Value'));
   currentMovieNum=handles.currentMovieNum; %count number of movie under
analysis
  tempRawData=handles.tempRawData;
%plot

   plotImage(handles.axes1,tempRawData{currentMovieNum}(:,:,val));

   if get(handles.getFirstFrameWithSpotsFromSlider,'Value');
      set(handles.firstFrameWithSpots,'String',num2str(val));
   end
   guidata(hObject, handles);
```

```
% --- Executes during object creation, after setting all properties.
function sliderChangeFrameInPlot_CreateFcn(hObject, eventdata, handles)
% hObject    handle to sliderChangeFrameInPlot (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor',[.9 .9 .9]);
end

function firstFrameWithSpots_Callback(hObject, eventdata, handles)
% hObject    handle to firstFrameWithSpots (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of firstFrameWithSpots as text
%      str2double(get(hObject,'String')) returns contents of firstFrameWithSpots as a double
input=str2double(get(hObject,'String'));
if input<1
   msgbox('Please input a positive number');
   return;
end
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function firstFrameWithSpots_CreateFcn(hObject, eventdata, handles)
% hObject    handle to firstFrameWithSpots (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in getFirstFrameWithSpotsFromSlider.
function getFirstFrameWithSpotsFromSlider_Callback(hObject, eventdata, handles)
% hObject    handle to getFirstFrameWithSpotsFromSlider (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of getFirstFrameWithSpotsFromSlider
```

```
% --- Executes on button press in previousMovie.
function previousMovie_Callback(hObject, eventdata, handles)
% hObject    handle to previousMovie (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%share data
if handles.currentMovieNum>1
    tempRawData=handles.tempRawData;
    handles.currentMovieNum=handles.currentMovieNum-1;
 plotImage(handles.axes1,tempRawData{handles.currentMovieNum}(:,:,1));

 %reset sothat firstImageWithSpots will be updated by frameNumSlider
   set(handles.getFirstFrameWithSpotsFromSlider,'Value',1);
   set(handles.sliderChangeFrameInPlot,'Value',5);
   set(handles.firstFrameWithSpots,'String','5');


currMovDis=[num2str(handles.currentMovieNum),'/',num2str(handles.totalNumMovies)]
;
   set(handles.currentMovieNumDisp,'String',currMovDis);
guidata(hObject,handles);
end

% --- Executes on button press in nextMovie.
function nextMovie_Callback(hObject, eventdata, handles)
% hObject    handle to nextMovie (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


if handles.currentMovieNum<handles.totalNumMovies
    tempRawData=handles.tempRawData;
    handles.currentMovieNum=handles.currentMovieNum+1;

%plot

plotImage(handles.axes1,tempRawData{handles.currentMovieNum}(:,:,1));
 %share data

 %reset sothat firstImageWithSpots will be updated by frameNumSlider
   set(handles.getFirstFrameWithSpotsFromSlider,'Value',1);
    set(handles.firstFrameWithSpots,'String','5');
   set(handles.sliderChangeFrameInPlot,'Value',5);
```

```
currMovDis=[num2str(handles.currentMovieNum),'/',num2str(handles.totalNumMovies)]
;
  set(handles.currentMovieNumDisp,'String',currMovDis);

   guidata(hObject,handles);
end

% --- Executes on button press in findPeaks.
function findPeaks_Callback(hObject, eventdata, handles)
% hObject    handle to findPeaks (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%find peaks of acceptor channel

   set(handles.getFirstFrameWithSpotsFromSlider,'Value',0);
   firstFrameWithSpots=str2double(get(handles.firstFrameWithSpots,'String'));

   tempRawData=handles.tempRawData;
%     fullFileName=handles.fullFileName;
   currentMovieNum=handles.currentMovieNum; %count number of movie under
analysis
   combIm=tempRawData{currentMovieNum}(:,:,firstFrameWithSpots);
%     combIm=imtophat(combIm,strel('disk',10));
   frams=str2double(get(handles.NumOfFramesAddToFindPeaks,'String'));
   for i=firstFrameWithSpots+1:firstFrameWithSpots+frams-1
%        aframe=imtophat(tempRawData{currentMovieNum}(:,:,i),strel('disk',10));
      aframe=tempRawData{currentMovieNum}(:,:,i);
      combIm=combIm+aframe;
   end
%background subtraction from the images

   combIm=imtophat(combIm,strel('disk',10));

%get peaks from registered image and plot in both channels.
      Mean=mean(combIm(:));
      STD=std(double(combIm(:)));
      stdInput=str2double(get(handles.stdInput,'String'));
      threshold=ceil(Mean+STD*stdInput);
      imageEdge=str2double(get(handles.ImageEdgeSize,'String'));
      [peakSet,roiDiaMat]=getImagePeaks(combIm,threshold,imageEdge);
%        [peakSet,roiDiaMat]=getCentOfMassCoor(combIm,peakSet);
%        figure; hist(roiDiaMat);

%update plot

      plotImage(handles.axes1,combIm,peakSet);
```

```
        set(handles.numOfSpotsFound,'String',num2str(size(peakSet,1)));

%update the slider
    maxThreshold=threshold*5;
    minThreshold=1;
    stepSize=[25,25]/(maxThreshold-minThreshold);
%                                   set(handles.sliderCombinedImageThreshold,        'SliderStep',
stepSize,'Min',minThreshold,'Max',maxThreshold,'Value',threshold);
%     set(handles.combinedImageThresholdDisplay,'String',num2str(threshold));
%share data
    handles.combIm{currentMovieNum}=combIm;
    handles.firstFrameWithSpotsMat{currentMovieNum}=firstFrameWithSpots;
    handles.peakSets{currentMovieNum}=peakSet;
    handles.roiDiaMats{currentMovieNum}=roiDiaMat;
    handles.thresholdValueMat{currentMovieNum}=threshold;
    handles.stdInputMat{currentMovieNum}=stdInput;

guidata(hObject, handles);

% --- Executes on slider movement.
function sliderCombinedImageThreshold_Callback(hObject, eventdata, handles)
% hObject    handle to sliderCombinedImageThreshold (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
    threshold=ceil(get(hObject,'Value'));
    currentMovieNum=handles.currentMovieNum;
        halfEdge=num2double(get(handles.ImageEdgeSize,'String'));
       [peakSet,roiDiaMat]=getImagePeaks(combIm,threshold,2*halfEdge);
%      [peakSet,roiDiaMat]=getCentOfMassCoor(combIm,peakSet);
    handles.roiDiaMats{currentMovieNum}=roiDiaMat;
    %plot data and peaks

    plotImage(handles.axes1,handles.combIm{currentMovieNum},peakSet);
    handles.peakSets{currentMovieNum}=peakSet;
    handles.thresholdValueMat{currentMovieNum}=threshold;
     set(handles.combinedImageThresholdDisplay,'String',num2str(threshold));
    set(handles.numOfSpotsFound,'String',num2str(size(peakSet,1)));
    guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function sliderCombinedImageThreshold_CreateFcn(hObject, eventdata, handles)
% hObject    handle to sliderCombinedImageThreshold (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor',[.9 .9 .9]);
end

function combinedImageThresholdDisplay_Callback(hObject, eventdata, handles)
% hObject    handle to combinedImageThresholdDisplay (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of combinedImageThresholdDisplay as text
%        str2double(get(hObject,'String')) returns contents of
combinedImageThresholdDisplay as a double
%     hWait=waitbar(0,'Pleas wait');
   threshold=round(str2double(get(hObject,'String')));
   currentMovieNum=handles.currentMovieNum;
       halfEdge=num2double(get(handles.ImageEdgeSize,'String'));
      [peakSet,roiDiaMat]=getImagePeaks(combIm,threshold,2*halfEdge);
   handles.roiDiaMats{currentMovieNum}=roiDiaMat;
   %plot data and peaks
  plotImage(handles.axes1,handles.combIm{currentMovieNum},peakSet);
   handles.peakSets{currentMovieNum}=peakSet;
   handles.thresholdValueMat{currentMovieNum}=threshold;

   set(handles.combinedImageThreshold,'Value',threshold);
   set(handles.numOfSpotsFound,'String',num2str(size(peakSet,1)));
%     close (hWait);
   guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function combinedImageThresholdDisplay_CreateFcn(hObject, eventdata, handles)
% hObject    handle to combinedImageThresholdDisplay (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if                ispc                &&                isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

function numOfSpotsFound_Callback(hObject, eventdata, handles)
% hObject    handle to numOfSpotsFound (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of numOfSpotsFound as text
%        str2double(get(hObject,'String')) returns contents of numOfSpotsFound as a double

% --- Executes on button press in populateData.
function populateData_Callback(hObject, eventdata, handles)
% hObject    handle to populateData (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%     message = [{'It will take a while'}; {'Working on file '};{num2str(1)}];
%     h=waitbar(0,message);

   totalNumMovies=handles.totalNumMovies;

   for i=1:totalNumMovies
%                         message = [{'It will take a while'}; {'Working on file
'};{[num2str(i),'/',num2str(totalNumMovies)]}];
%         waitbar(i/totalNumMovies,h,message)
       if ~isempty(handles.peakSets{i}) %if peakset is not empty
       populateAMovieData(handles.fullFileName{i},handles.peakSets{i},...
          handles.firstFrameWithSpotsMat{i},...
          handles.combIm{i},handles,i);
       end
   end
%       close(h);

guidata(hObject, handles);

function
populateAMovieData(fullFileName,peakSet,firstFrameWithSpots,combIm,handles,curre
ntMovieNum)

 %load the movie

       stdInput=handles.stdInputMat{currentMovieNum};
       MovieInfo = imfinfo(fullFileName);
       width=MovieInfo.Width;
       height=MovieInfo.Height;
       totalFrameNum = length(MovieInfo);
       rawData = zeros(height,width,totalFrameNum,'uint16');
       hp=waitbar(0,['Reading Movie # ',num2str(currentMovieNum)]);
      alreadyLoadedDAta=handles.tempRawData{currentMovieNum};
      alreadyLoadedDAtaLength=size(alreadyLoadedDAta,3);
```

```
      rawData(:,:,1:alreadyLoadedDAtaLength)=alreadyLoadedDAta;
       for k = alreadyLoadedDAtaLength+1:totalFrameNum
          updatWaitbar=[{['Reading Movie # ',num2str(currentMovieNum)]};
{[num2str(k),'/',num2str(totalFrameNum)]}];
          waitbar(k/totalFrameNum,hp,updatWaitbar);
          rawData(:,:,k)=imread(fullFileName, k, 'Info', MovieInfo);
       end
       close(hp);

       peakNum=size(peakSet,1);
       outputData=zeros(totalFrameNum,peakNum);

          Co=                                                    getRoiCoord_variableRois(combIm,
peakSet,str2double(get(handles.roiLength,'String')));%get bright pixels locations as [x1 x2
y1 y2]= [c1 c2 r1 r2], x1=smallest x and x2=highest x
          roiLengthArray=
str2double(get(handles.roiLength,'String'))*ones(length(peakSet),1);%for saving data

       bCo= getBackgndRoiCord(Co,combIm);
       hF=waitbar(0,['Generating Time Traces of Movie # ',num2str(currentMovieNum)]);

           if get(handles.backgndSubtrctn_ringAroundRoi,'Value')
             for frame =firstFrameWithSpots:totalFrameNum %keep all frames before
molecules appear to be zero
                 waitbar(frame/totalFrameNum,hF,[{['Generating Time Traces of Movie #
',num2str(currentMovieNum)]};{[num2str(frame),'/',num2str(totalFrameNum)]}]);
               for peak=1:peakNum;
                  Roi=rawData(Co(peak,3):Co(peak,4),Co(peak,1):Co(peak,2),frame);

backRoi=rawData(bCo(peak,3):bCo(peak,4),bCo(peak,1):bCo(peak,2),frame);
                  backRoi=[backRoi(:,1)',backRoi(:,end)',backRoi(1,2:end-
1),backRoi(end,2:end-1)]; %only edges are background
                  backRoi=sort(backRoi(:));
                  backRoi=mean(backRoi(1:round(0.75*length(backRoi))));
                  meanRoi=mean(Roi(:))-backRoi;
                  outputData(frame,peak)=meanRoi;
               end
             end

           else

             for frame =firstFrameWithSpots:totalFrameNum %keep all frames before
molecules appear to be zero
               waitbar(frame/totalFrameNum,hF,[{['Generating Time Traces of Movie #
',num2str(currentMovieNum)]};{[num2str(frame),'/',num2str(totalFrameNum)]}]);
               Im=rawData(:,:,frame);
```

```
            Im=imtophat(Im,strel('disk',10));
             for peak=1:peakNum;
                Roi=Im(Co(peak,3):Co(peak,4),Co(peak,1):Co(peak,2));
                meanRoi=mean(Roi(:));
                outputData(frame,peak)=meanRoi;
             end
          end
        end


    close (hF);

%if any value is negative
       for i=1:size(outputData,2)
         Min=min(outputData(:,i));
         if Min<0 ;
            outputData(:,i)=outputData(:,i)+ abs(Min);
         end
       end

%remove column with junck only

      newOutPutData=[];
      newPeakSet=[];
      newRoiLengthArray=[];
       for i=1:size(outputData,2)
          mean_1st20 =mean(outputData(firstFrameWithSpots:firstFrameWithSpots+19,i));
          mean_last50=mean(outputData(end-49:end,i));
          STD_last50=std(outputData(end-49:end,i));
          if (mean_1st20-mean_last50)>STD_last50 *3
             newOutPutData=[newOutPutData,outputData(:,i)];
             newPeakSet=[newPeakSet;peakSet(i,:)];
             newRoiLengthArray=[newRoiLengthArray;roiLengthArray(i,:)];
          end
       end
       outputData=newOutPutData;
       peakSet=newPeakSet;
       roiLengthArray=newRoiLengthArray;
       % roiDia matrix

%add an column for frame number
       newCol=(1:size(outputData,1))';
       outputData=[newCol,outputData];
       intiallyTotalPeaksFound=peakNum;
       condition2removeGarbage='Garbage removal condition : mean of first 20 frames after
molecule appear on movie - mean of last 50 frames > 3 * std of last 50 frames';
```

```
    saveTimeTraces_variableRoi(outputData,fullFileName,peakSet,roiLengthArray);

NumberOfSpotManuallySelected=handles.NumOfPeaksManuallySelected(currentMovie
Num,1);

getLog(handles,fullFileName,peakSet,firstFrameWithSpots,NumberOfSpotManuallySele
cted,stdInput,intiallyTotalPeaksFound,condition2removeGarbage)

function
getLog(handles,fileName,peakSet,firstFrameWithSpots,NumberOfSpotManuallySelected
,stdInput,intiallyTotalPeaksFound,condition2removeGarbage)

    hwaitbar=waitbar(0,'Saving Log file');
        LogData=cell(15,1);
        [~,softwareName,~]=fileparts(mfilename('fullpath'));
        d = dir([mfilename('fullpath'),'.m']);
        moddate = d.date;
        [pathName,fileName,~] = fileparts(fileName) ;
        time=['Analysis done on : ',datestr(clock, 0)];
        firstFrame=['first frame where molecule appeared : ',num2str(firstFrameWithSpots)];
        stdInput=['Condition to find peaks: mean + ',num2str(stdInput),'*STD'];
        NumberOfSpotInitiallyFound=['Total spots intially Found :
',num2str(intiallyTotalPeaksFound)];
        NumberOfSpot=['Total spots was kept and saved after garbage removal :
',num2str(size(peakSet,1))];

            roiLength=get(handles.roiLength,'String');
            roiLength=['Roi lenght : ', roiLength];
        if get(handles.backgndSubtrctn_movingDisk,'Value')
            backgroundSub=['Background subtraction type :','moving disk with dia of 10
pixel'];
        else
            backgroundSub=['Background subtraction type :','mean of the ring around (roi
length + 4 pixel diameter)'];
        end

frameNumtoCombinedImage=get(handles.NumOfFramesAddToFindPeaks,'String');

NumberOfSpotManuallySelected=['NumberOfSpotManuallySelected',num2str(NumberO
fSpotManuallySelected)];

        i=1;
        LogData{i}=['Software used : ',softwareName, '  last modified on ',moddate];
        i=i+1;
        LogData{i}=time;
        i=i+1;
```

```
        LogData{i}=['Movie name : ',fileName];
        i=i+1;
        LogData{i}=firstFrame;
        i=i+1;
        LogData{i}=['Number of frames added to make combined image : '
frameNumtoCombinedImage];
        i=i+1;
        LogData{i}=stdInput;
        i=i+1;
        LogData{i}=NumberOfSpotInitiallyFound;
        i=i+1;
        LogData{i}= NumberOfSpotManuallySelected;
        i=i+1;
        LogData{i}=NumberOfSpot;
        i=i+1;
        LogData{i}=roiLength;
        i=i+1;
           LogData{i}=backgroundSub;
        i=i+1;
        LogData{i}=condition2removeGarbage;

        fileName=['Log of ',fileName,'.txt'];
        newFileName=fullfile(pathName,fileName);

        %same the matrix as .txt file remember file name already has an extenstion
        %of .txt
        fileID = fopen(newFileName, 'wt');
%          fprintf(fileID,'%s \n',LogData.');
        [nrows,~] = size(LogData);
        for row = 1:nrows
           fprintf(fileID,'%s \n',LogData{row,:});
        end
        fclose(fileID);
% dlmwrite('my_data.out',A, ';')

         close(hwaitbar);

function roiLength_Callback(hObject, eventdata, handles)
% hObject    handle to roiLength (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of roiLength as text
%        str2double(get(hObject,'String')) returns contents of roiLength as a double
```

```
% --- Executes during object creation, after setting all properties.
function roiLength_CreateFcn(hObject, eventdata, handles)
% hObject    handle to roiLength (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

function NumOfFramesAddToFindPeaks_Callback(hObject, eventdata, handles)
% hObject    handle to NumOfFramesAddToFindPeaks (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of NumOfFramesAddToFindPeaks as text
%        str2double(get(hObject,'String')) returns contents of NumOfFramesAddToFindPeaks as a double
input=str2double(get(hObject,'String')) ;

if input<2
   set(hObject,'String','2')
end
guidata(hObject, handles);

function stdInput_Callback(hObject, eventdata, handles)
% hObject    handle to stdInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of stdInput as text
%        str2double(get(hObject,'String')) returns contents of stdInput as a double


% --- Executes during object creation, after setting all properties.
function stdInput_CreateFcn(hObject, eventdata, handles)
% hObject    handle to stdInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in backgndSubtrctn_ringAroundRoi.
function backgndSubtrctn_ringAroundRoi_Callback(hObject, eventdata, handles)
% hObject    handle to backgndSubtrctn_ringAroundRoi (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of backgndSubtrctn_ringAroundRoi

if get(hObject,'Value');
    set(handles.backgndSubtrctn_movingDisk,'Value',0);
else
    set(handles.backgndSubtrctn_movingDisk,'Value',1);
end

guidata(hObject, handles);

% --- Executes on button press in backgndSubtrctn_movingDisk.
function backgndSubtrctn_movingDisk_Callback(hObject, eventdata, handles)
% hObject    handle to backgndSubtrctn_movingDisk (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of backgndSubtrctn_movingDisk

if get(hObject,'Value');
    mes='This condition will use a "imtophat" function with diameter of 10 pixels. Make sure you know that this condition meet your requirement';
    msgbox(mes,'Warning');
    set(handles.backgndSubtrctn_ringAroundRoi,'Value',0);
else
    set(handles.backgndSubtrctn_ringAroundRoi,'Value',1);
end

guidata(hObject, handles);

% --- Executes on button press in addAPeak.
function addAPeak_Callback(hObject, eventdata, handles)
% hObject    handle to addAPeak (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[xc,yc] = ginput(1);
```

```
currentMovieNum=handles.currentMovieNum;
framLength=size(handles.combIm{currentMovieNum},1);
framWidth=size(handles.combIm{currentMovieNum},2);
if xc>10 && yc>10 && xc<framWidth-10 && yc<framLength-10
   [aPeakCo,~]=getClosestPeakCenter(handles.combIm{currentMovieNum},[xc,yc]);
   [aPeakCo,aDia]=getCentOfMassCoor(handles.combIm{currentMovieNum},aPeakCo);

handles.peakSets{currentMovieNum}=[handles.peakSets{currentMovieNum};aPeakCo];

handles.roiDiaMats{currentMovieNum}=[handles.roiDiaMats{currentMovieNum};aDia]
;
   plotImage(handles.axes1,
handles.combIm{currentMovieNum},handles.peakSets{currentMovieNum})
%plotImage(axesLoc,d,peakSet)
        set(handles.numOfSpotsFound,'String',num2str(size(handles.peakSets{currentMo
vieNum},1)));

handles.NumOfPeaksManuallySelected(currentMovieNum,1)=handles.NumOfPeaksMan
uallySelected(currentMovieNum,1)+1;
else
   msgbox('Opps! Looks like your selected peak is out of range. Remember: 10 pixels from
any edge cannot be selected','Error','error');
end
guidata(hObject, handles);

% --- Executes on button press in deleteLastPeak.
function deleteLastPeak_Callback(hObject, eventdata, handles)
% hObject    handle to deleteLastPeak (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
   currentMovieNum=handles.currentMovieNum;
   handles.peakSets{currentMovieNum}(end,:)=[];
   handles.roiDiaMats{currentMovieNum}(end,:)=[];
   plotImage(handles.axes1,
handles.combIm{currentMovieNum},handles.peakSets{currentMovieNum})
%plotImage(axesLoc,d,peakSet)
        set(handles.numOfSpotsFound,'String',num2str(size(handles.peakSets{currentMo
vieNum},1)));

handles.NumOfPeaksManuallySelected(currentMovieNum,1)=handles.NumOfPeaksMan
uallySelected(currentMovieNum,1)-1;
   guidata(hObject, handles);


% --- Executes on button press in zoomAPeak.
function zoomAPeak_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to zoomAPeak (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
      currentMovieNum=handles.currentMovieNum;
      [xc,yc] = ginput(1);
      clickedAx = gca;
      combIm=handles.combIm{currentMovieNum};
      if isempty(combIm)
         firstFrameWithSpots=str2double(get(handles.firstFrameWithSpots,'String'));
         tempRawData=handles.tempRawData;
         currentMovieNum=handles.currentMovieNum; %count number of movie under
analysis
         combIm=tempRawData{currentMovieNum}(:,:,firstFrameWithSpots);
      end
      multiplier=round(size(combIm,1)/512); %to roi size selection
      roiD=12*multiplier; %if the size of frame is 512, then roi dia is 16, if size if larger roi
dia gets multiplied
      framLength=size(combIm,1);
      framWidth=size(combIm,2);
      if xc>10 && yc>10 && xc<framWidth-10 && yc<framLength-10
%           [aPeakCo,~]=getClosestPeakCenter(combIm,[xc,yc]);
         [aPeakCo,~]=getCentOfMassCoor(combIm,[xc,yc]);

[aDia]=CorrespondingDia(handles.peakSets{currentMovieNum},handles.roiDiaMats{cur
rentMovieNum}, aPeakCo);
          xRoiEdge=round(aPeakCo(:,1))-roiD/2;
         yRoiEdge=round(aPeakCo(:,2))-roiD/2;
         aRoi=combIm(yRoiEdge+1:yRoiEdge+roiD,xRoiEdge+1:xRoiEdge+roiD); % roi
         figure; bar3(aRoi);
% %           if ~isempty(aDia)
% %              Str=['Recorded diameter was ',num2str(aDia)];
% %              title(Str);
% % %                text(Str);
% %           else
% %              Str='Select the center of an already selected peak to get recorded diameter';
% %              title(Str);
% %           end

% [peakCentOut,diameterOut]=getCentOfMassCoor(combIm,aPeakCo)
      else
         msgbox('Opps! Looks like your selected peak is out of range. Remember: 10 pixels
from any edge cannot be selected','Error','error');
      end

guidata(hObject, handles);
```

```
function [aDia]=CorrespondingDia(allPeaks,allDia, aPeak)
%                                   temp=abs(allPeaks-[aPeak(1)*ones(size(allPeaks,1),1),
aPeak(2)*ones(size(allPeaks,1),1)]);

col1=allPeaks(:,1);
col2=allPeaks(:,2);
try
   temp3=logical((col1==aPeak(1)).*(col2==aPeak(2)));
   aDia=allDia(temp3);
catch ME
   aDia=[];
end
% aPeaksCoor=[col1(temp3),col2(temp3)];

function ImageEdgeSize_Callback(hObject, eventdata, handles)
% hObject    handle to ImageEdgeSize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ImageEdgeSize as text
%        str2double(get(hObject,'String')) returns contents of ImageEdgeSize as a double
 val=str2double(get(hObject,'String'));
 set(hObject,'String',num2str(round(val)));
  guidata(hObject, handles);


% --- Executes during object creation, after setting all properties.
function ImageEdgeSize_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ImageEdgeSize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if                  ispc                &&                  isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

function [peakSet,peakDiameters]=getImagePeaks(Im,thres,edge)
   peakSet=[];
   if nargin<3
      edge=10;
   end
```

```
   dx=floor(edge/2);
   for c =edge:dx:size(Im,2)-edge-dx
      for r=edge:dx:size(Im,1)-edge-dx
         aRoi=Im(r+1:r+dx,c+1:c+dx);
         if max(max(aRoi))>thres
            Max=max(aRoi(:));
            [rIdx,cIdx]=find(aRoi==Max);
           if length(rIdx)>1 || length(rIdx)>1
              cIdx=dx/2;
              rIdx=dx/2;

           end
            aPeak=[c,r]+[cIdx,rIdx];
            peakSet=[peakSet;aPeak];
         end
      end
   end
peakSet=double(round(peakSet));

[peakSet,peakDiameters]=getCentOfMassCoor_myfindpeaks(Im,peakSet,edge,thres);
%removes false peaks
   if  isempty(peakSet)
      msgbox('No Peak was found, lowering threshold might help to find some peaks')
     return;
   end
[peakSet,peakDiameters]=removeClosedPeaks_myfindPeaks(Im,peakSet,dx,peakDiamet
ers); %removes close peaks and avoid putting same peak multiple times

if  isempty(peakSet)
      msgbox('No Peak was found, lowering threshold might help to find some peaks')
     return;
end
   peakDiameters=ones(length(peakSet),1);

function
[peakSetOut,peakDiaOut]=removeClosedPeaks_myfindPeaks(Im,peakSet,dx,peakDiaIn)

      disThrshold=dx*1.14; %distace =dx*sqrt(2);
      peakSetA=peakSet;
      peakSetOut=[];
      peakDiaOut=[];
%        minDistMat=600*ones(size(peakSet,1),1);
    for i=1:size(peakSet,1)
       distMat=zeros(size(peakSet,1),1);
      for j=1:size(peakSetA,1)
            dist=sqrt((peakSet(i,1)-peakSetA(j,1))^2 +(peakSet(i,2)-peakSetA(j,2))^2);
```

```
            distMat(j)=dist;
        end
       if  numel(distMat(distMat<disThrshold))==1 %if one peak with lower than threshold
dist, accpt it
          peakSetOut=[peakSetOut;peakSet(i,:)];
          if nargin==4
             peakDiaOut=[peakDiaOut;peakDiaIn(i)];
          end
       else  %otherwise accept one with highest intensity on the image
          disputedPeaks=peakSet(distMat<disThrshold,:);
          disputedPeaksIndex=find(distMat<disThrshold);
          intesityDisputedPeaks=zeros(size(size(disputedPeaks,2),1));
          for k=1:size(disputedPeaks,2)
              intesityDisputedPeaks(k,1)=Im(disputedPeaks(k,2),disputedPeaks(k,1));
          end
          [~,Idx]=max(intesityDisputedPeaks);
          originalIndex=disputedPeaksIndex(Idx);
          if originalIndex==i %avoid putting same peaks multiple times
           peakSetOut=[peakSetOut;peakSet(originalIndex,:)];
           if nargin==4
             peakDiaOut=[peakDiaOut;peakDiaIn(originalIndex)];
           end

          end
       end
     end


function
[peakCentOut,diameterOut]=getCentOfMassCoor_myfindpeaks(Im,peakSets,edge,thres)

        peakCent=zeros(size(peakSets));
        diameterMat=zeros(size(peakSets,1),1);

%       multiplier=round(size(d,1)/512); %to roi size selection
       roiD=12; %if the size of frame is 512, then roi dia is 16, if size if larger roi dia gets
multiplied
        xRoiEdge=round(peakSets(:,1))-roiD/2;
        yRoiEdge=round(peakSets(:,2))-roiD/2;
     for i=1:size(peakSets,1)
       aRoi=Im(yRoiEdge(i)+1:yRoiEdge(i)+roiD,xRoiEdge(i)+1:xRoiEdge(i)+roiD);    %
roi

% % %       [centA,diaA, gof,outputData] = createFit_gaussian2D(double(aRoi));
% % % %         figure; surf(aRoi(:,1:10));
% % %        figure; bar3(aRoi);
```

```
% % %       Str=['Diameter ',num2str(diaA)];
% % %       legend(Str);
% % %       diameterMat(i,1)=round(diaA);
% % %       peakCent(i,:)=[xRoiEdge(i),yRoiEdge(i)]+round(centA);

          tempRoi=aRoi(:);
          tempMat=sort(tempRoi);
          tempMat=tempMat(1:round(0.75*length(tempMat)));
          tempThres=mean(tempMat)+3*std(double(tempMat));

       if nargin<4
          thres=tempThres;
       elseif thres>tempThres; %lowest value is the thereshould value
          thres=tempThres;
       end

       dn=aRoi.*uint16(aRoi>thres);
       stats = regionprops(logical(dn),dn,'Area','WeightedCentroid');
       stats=stats([stats.Area]>2);
       cents=[stats.WeightedCentroid]';
       cents=[cents(1:2:end),cents(2:2:end)];
          if isempty(cents); %no peaks is found
             peakCent(i,:)=peakSets(i,:);
             diameterMat(i,1)=0;
          elseif size(cents,1)==1 %onley one peak is found
              Idx=1;
              peakCent(i,:)=[xRoiEdge(i),yRoiEdge(i)]+round(cents(Idx,:));
              diameterMat(i,1)=round(sqrt(stats(Idx).Area));
%
diameterMat(i,1)=ceil((stats(Idx).MajorAxisLength+stats(Idx).MinorAxisLength)/2);
          else %if more than one center is present, select one closest to center of roi
              distMat=zeros(size(cents,1),1);
              roiCent=[roiD/2,roiD/2];
                 for j=1:size(cents,1);
                    dist=sqrt((cents(j,1)-roiCent(1,1))^2 +(cents(j,2)-roiCent(1,2))^2);
                    distMat(j)=dist;
                 end
              [~,Idx]=min(distMat);
             diameterMat(i,1)=round(sqrt(stats(Idx).Area));
             peakCent(i,:)=[xRoiEdge(i),yRoiEdge(i)]+round(cents(Idx,:));
%
diameterMat(i,1)=round((stats(Idx).MajorAxisLength+stats(Idx).MinorAxisLength)/2);
          end


%         figure; bar3(aRoi);
```

```
%         Str=['Diameter ',num2str(diameterMat(i,1))];
%         legend(Str);

    end

    %remove peaks with diameter less than 2, wich was recorded as 2
    peakCentOut=[];
    diameterOut=[];
    for i=1:size(diameterMat,1)
%               if diameterMat(i,1)~=0 && peakCent(i,1)>10 && peakCent(i,2)>10 &&
peakCent(i,1)<size(d,2)-10 && peakCent(i,2)<size(d,1)-10
       if diameterMat(i,1)>2 && diameterMat(i,1)<length(aRoi) && peakCent(i,1)>edge
&&        peakCent(i,2)>edge        &&        peakCent(i,1)<size(Im,2)-edge        &&
peakCent(i,2)<size(Im,1)-edge
           peakCentOut=[peakCentOut;peakCent(i,:)];
           diameterOut=[diameterOut;diameterMat(i,1)];
        end
    end
% % %      peakCentOut=peakCent;
% % %           diameterOut=diameterMat;


function saveTimeTraces_variableRoi(outputData,fullFileName,peakSet,roiLengthArray)

   hWaitBar=waitbar(0,'Please wait, saving data as Excell file');

colheaders=cell(1,size(outputData,2));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%if you dont put anything at colheaders{1}, matlab cannot detect colheaders
%in the structure file
colheaders{1}='Frame Number';

for i=2:size(outputData,2);
colheaders{i}=[num2str(i-1),'                         roiDia-',num2str(roiLengthArray(i-1)),'(',
num2str(peakSet(i-1,1)),',',num2str(peakSet(i-1,2)),')'];
end

outputData=num2cell(outputData);

if strcmp(fullFileName(end-3:end),'xlsx')==1;
   fullFileName=fullFileName(1:end-5);%remove .xlsx from the file name
else
   fullFileName=fullFileName(1:end-4);
end
newFileName=[fullFileName,'.xlsx'];
```

```
%if a file with this name already exist, delete it
   if exist(newFileName,'file')==2
      delete(newFileName);
   end
outputData=[colheaders; outputData];
if ismac()
   my_xlwrite(newFileName,outputData,'Sheet1', 'A1')
else
   xlswrite(newFileName,outputData);
end
close(hWaitBar);

function plotImage(axesLoc,Im,peakSet)
axes(axesLoc);
% imagesc(imadjust(d), 'parent', axesLoc);
imagesc(Im, 'parent', axesLoc);
if nargin==3 && isempty(peakSet)
   msgbox('No Peak to plot', 'Error','error');
elseif nargin==3
   hold on
   x=peakSet(:,1);
   y=peakSet(:,2);
   plot(x,y,'ro');
   %label the spots as 1,2,3,...
   labelOfPoints=(1:size(peakSet,1))';
   b = num2str(labelOfPoints);
   c = cellstr(b);
   dx = 0.1; dy = 0.1; % displacement so the text does not overlay the data poi
   text(x+dx, y+dy, c);
   hold off
end

% colormap('gray');
set(axesLoc,'XTickLabel','','YTickLabel','','xtick',[],'ytick',[]);

function [peakCentOut,diameterOut]=getClosestPeakCenter(d,peakSets,thres)
      %get a 10*10 region to ger roi center and diamters
      peakCentOut=zeros(size(peakSets));
      diameterOut=zeros(size(peakSets,1),1);
%        xVal=round(peakSets(:,1)); yVal=round(peakSets(:,2));

     multiplier=round(size(d,1)/512);
     roiD=10*multiplier; %is the size of frame is 512, then roi dia is 10, if size if larger roi
dia gets multiplied
      edge_xVal=round(peakSets(:,1))-roiD/2;
      edge_yVal=round(peakSets(:,2))-roiD/2;
```

```
    for i=1:size(peakSets,1)
      aRoi=d(edge_yVal(i)+1:edge_yVal(i)+roiD,edge_xVal(i)+1:edge_xVal(i)+roiD);  %
row = y value and col = x val.
      if nargin<3
%           thres = multithresh(aRoi);
      thres=mean(aRoi(:));
      end

      dn=aRoi.*uint16(aRoi>thres);
      stats                                                                    =
regionprops(logical(dn),dn,'Area','WeightedCentroid','MajorAxisLength','MinorAxisLeng
th');
      stats=stats([stats.Area]>1);
      cents=[stats.WeightedCentroid]';
      cents=[cents(1:2:end),cents(2:2:end)];
        if isempty(cents); %no peaks is found
          peakCentOut(i,:)=peakSets(i,:);
          diameterOut(i,1)=2;
        elseif size(cents,1)==1 %onley one peak is found
          Idx=1;
          peakCentOut(i,:)=[edge_xVal(i),edge_yVal(i)]+round(cents(Idx,:));

diameterOut(i,1)=ceil((stats(Idx).MajorAxisLength+stats(Idx).MinorAxisLength)/2);
        else %if more than one center is present, select one closest to center of roi
          distMat=zeros(size(cents,1),1);
          roiCent=[roiD/2,roiD/2];
            for j=1:size(cents,1);
              dist=sqrt((cents(j,1)-roiCent(1,1))^2 +(cents(j,2)-roiCent(1,2))^2);
              distMat(j)=dist;
            end
          [~,Idx]=min(distMat);
          peakCentOut(i,:)=[edge_xVal(i),edge_yVal(i)]+round(cents(Idx,:));

diameterOut(i,1)=ceil((stats(Idx).MajorAxisLength+stats(Idx).MinorAxisLength)/2);
        end
    end

    %check if any peaks values goes beyound the edge of image
    for i=1:size(peakSets,1)
       if peakCentOut(i,1)<10 && peakCentOut(i,2)<10 && peakCentOut(i,1)>size(d,2)-
10 && peakCentOut(i,2)>size(d,1)-10
         if peakCentOut(i,1)<10
            peakCentOut(i,1)=10;
         elseif peakCentOut(i,2)<10
           peakCentOut(i,2)=10;
         elseif peakCentOut(i,1)>size(d,2)-10
```

```
            peakCentOut(i,1)=size(d,2)-10;
        elseif peakCentOut(i,2)>size(d,1)-10
            peakCentOut(i,1)=size(d,1)-10;
        end
    end
end


function [peakCentOut,diameterOut]=getCentOfMassCoor(d,peakSets,thres)

    %get a 10*10 region to ger roi center and diamters
    peakCent=zeros(size(peakSets));
    diameterMat=zeros(size(peakSets,1),1);
%       xVal=round(peakSets(:,1)); yVal=round(peakSets(:,2));

    multiplier=round(size(d,1)/512); %to roi size selection
    roiD=12*multiplier; %if the size of frame is 512, then roi dia is 16, if size if larger roi
dia gets multiplied
    xRoiEdge=round(peakSets(:,1))-roiD/2;
    yRoiEdge=round(peakSets(:,2))-roiD/2;
for i=1:size(peakSets,1)
    aRoi=d(yRoiEdge(i)+1:yRoiEdge(i)+roiD,xRoiEdge(i)+1:xRoiEdge(i)+roiD); % roi

% % %       [centA,diaA, gof,outputData] = createFit_gaussian2D(double(aRoi));
% % % %         figure; surf(aRoi(:,1:10));
% % %        figure; bar3(aRoi);
% % %        Str=['Diameter ',num2str(diaA)];
% % %        legend(Str);
% % %        diameterMat(i,1)=round(diaA);
% % %        peakCent(i,:)=[xRoiEdge(i),yRoiEdge(i)]+round(centA);

    if nargin<3
        tempRoi=aRoi(:);
        tempMat=sort(tempRoi);
        tempMat=tempMat(1:round(0.80*length(tempMat)));
        thres=mean(tempMat)+3*std(double(tempMat));
    end

    dn=aRoi.*uint16(aRoi>thres);
    stats = regionprops(logical(dn),dn,'Area','WeightedCentroid');
    stats=stats([stats.Area]>2);
    cents=[stats.WeightedCentroid]';
    cents=[cents(1:2:end),cents(2:2:end)];
        if isempty(cents); %no peaks is found
            peakCent(i,:)=peakSets(i,:);
            diameterMat(i,1)=0;
```

```
        elseif size(cents,1)==1 %onley one peak is found
            Idx=1;
            peakCent(i,:)=[xRoiEdge(i),yRoiEdge(i)]+round(cents(Idx,:));
            diameterMat(i,1)=round(sqrt(stats(Idx).Area));
%
diameterMat(i,1)=ceil((stats(Idx).MajorAxisLength+stats(Idx).MinorAxisLength)/2);
        else %if more than one center is present, select one closest to center of roi
            distMat=zeros(size(cents,1),1);
            roiCent=[roiD/2,roiD/2];
                for j=1:size(cents,1);
                    dist=sqrt((cents(j,1)-roiCent(1,1))^2 +(cents(j,2)-roiCent(1,2))^2);
                    distMat(j)=dist;
                end
            [~,Idx]=min(distMat);
           diameterMat(i,1)=round(sqrt(stats(Idx).Area));
           peakCent(i,:)=[xRoiEdge(i),yRoiEdge(i)]+round(cents(Idx,:));
%
diameterMat(i,1)=round((stats(Idx).MajorAxisLength+stats(Idx).MinorAxisLength)/2);
        end


%       figure; bar3(aRoi);
%       Str=['Diameter ',num2str(diameterMat(i,1))];
%       legend(Str);

    end

    %remove peaks with diameter less than 2, wich was recorded as 2
    peakCentOut=[];
    diameterOut=[];
    for i=1:size(diameterMat,1)
%               if diameterMat(i,1)~=0 && peakCent(i,1)>10 && peakCent(i,2)>10 &&
peakCent(i,1)<size(d,2)-10 && peakCent(i,2)<size(d,1)-10
      if diameterMat(i,1)>1 && diameterMat(i,1)<length(aRoi) && peakCent(i,1)>10 &&
peakCent(i,2)>10 && peakCent(i,1)<size(d,2)-10 && peakCent(i,2)<size(d,1)-10
          peakCentOut=[peakCentOut;peakCent(i,:)];
          diameterOut=[diameterOut;diameterMat(i,1)];
       end
    end
% % %     peakCentOut=peakCent;
% % %          diameterOut=diameterMat;


function [outputMatrix] = getRoiCoord_variableRois(d, peakSet,roiDiameterArray)

if size(roiDiameterArray,1)==1; %roi size is not variable
```

```
        RoiLength=roiDiameterArray;
         if RoiLength==1
            [outputMatrix] = get1by1RoiCoor(peakSet);
        elseif RoiLength==2
            [outputMatrix] = get2by2RoiCoor(d, peakSet);
        elseif  RoiLength==3
            [outputMatrix] = get2by2RoiCoor(d, peakSet);
            [outputMatrix] = get3by3RoiCoor(d, outputMatrix);
        elseif RoiLength==4
            [outputMatrix] = get2by2RoiCoor(d, peakSet);
            [outputMatrix] = get3by3RoiCoor(d, outputMatrix);
            [outputMatrix] = get4by4RoiCoor(d, outputMatrix);
       elseif RoiLength==5
            [outputMatrix] = get2by2RoiCoor(d, peakSet);
            [outputMatrix] = get3by3RoiCoor(d, outputMatrix);
            [outputMatrix] = get4by4RoiCoor(d, outputMatrix);
            [outputMatrix] = get5by5RoiCoor(d, outputMatrix);
       elseif RoiLength==6
            [outputMatrix] = get2by2RoiCoor(d, peakSet);
            [outputMatrix] = get3by3RoiCoor(d, outputMatrix);
            [outputMatrix] = get4by4RoiCoor(d, outputMatrix);
            [outputMatrix] = get5by5RoiCoor(d, outputMatrix);
            [outputMatrix] = get6by6RoiCoor(d, outputMatrix);

        elseif RoiLength>6 && mod(RoiLength,2)==1;%odd 7,9,11...
            dx=RoiLength-5; %2,4,6...
             [outputMatrix] = get2by2RoiCoor(d, peakSet);
            [outputMatrix] = get3by3RoiCoor(d, outputMatrix);
            [outputMatrix] = get4by4RoiCoor(d, outputMatrix);
            [outputMatrix] = get5by5RoiCoor(d, outputMatrix);
            [outputMatrix] = increaseRoiby2Npixels(outputMatrix,dx);
        elseif RoiLength>6 && mod(RoiLength,2)==0; %even 8,10,12...
            dx=RoiLength-6; %2,4,6...
            [outputMatrix] = get2by2RoiCoor(d, peakSet);
            [outputMatrix] = get3by3RoiCoor(d, outputMatrix);
            [outputMatrix] = get4by4RoiCoor(d, outputMatrix);
            [outputMatrix] = get5by5RoiCoor(d, outputMatrix);
            [outputMatrix] = get6by6RoiCoor(d, outputMatrix);
            [outputMatrix] = increaseRoiby2Npixels(outputMatrix,dx);

         end

else
        outputMatrix=[];
        for i=1:length(roiDiameterArray)
          RoiLength= roiDiameterArray(i);
```

```
        aPeak=peakSet(i,:);

         if RoiLength==1
            [tempoutputMatrix] = get1by1RoiCoor(aPeak);
         elseif RoiLength==2
            [tempoutputMatrix] = get2by2RoiCoor(d, aPeak);
         elseif  RoiLength==3
            [tempoutputMatrix] = get2by2RoiCoor(d, aPeak);
            [tempoutputMatrix] = get3by3RoiCoor(d, tempoutputMatrix);
         elseif RoiLength==4
            [tempoutputMatrix] = get2by2RoiCoor(d, aPeak);
            [tempoutputMatrix] = get3by3RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get4by4RoiCoor(d, tempoutputMatrix);
        elseif RoiLength==5
            [tempoutputMatrix] = get2by2RoiCoor(d, aPeak);
            [tempoutputMatrix] = get3by3RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get4by4RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get5by5RoiCoor(d, tempoutputMatrix);
        elseif RoiLength==6
            [tempoutputMatrix] = get2by2RoiCoor(d, aPeak);
            [tempoutputMatrix] = get3by3RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get4by4RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get5by5RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get6by6RoiCoor(d, tempoutputMatrix);

         elseif RoiLength>6 && mod(RoiLength,2)==1;%odd 7,9,11...
            dx=RoiLength-5; %2,4,6...
             [tempoutputMatrix] = get2by2RoiCoor(d, aPeak);
            [tempoutputMatrix] = get3by3RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get4by4RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get5by5RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = increaseRoiby2Npixels(tempoutputMatrix,dx);
         elseif RoiLength>6 && mod(RoiLength,2)==0; %even 8,10,12...
            dx=RoiLength-6; %2,4,6...
            [tempoutputMatrix] = get2by2RoiCoor(d, aPeak);
            [tempoutputMatrix] = get3by3RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get4by4RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get5by5RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = get6by6RoiCoor(d, tempoutputMatrix);
            [tempoutputMatrix] = increaseRoiby2Npixels(tempoutputMatrix,dx);

         end
         outputMatrix=[outputMatrix;tempoutputMatrix];
      end
end
```

```
function [outputMatrix] = increaseRoiby2Npixels(inMat,dx)

      outputMatrix=inMat;
      outputMatrix(:,1)=inMat(:,1)-dx/2; %new x1=old x1-dx/2
      outputMatrix(:,2)=inMat(:,2)+dx/2;%new x2=old x2+dx/2
      outputMatrix(:,3)=inMat(:,3)-dx/2;
      outputMatrix(:,4)=inMat(:,4)+dx/2;



function [outputMatrix] = get1by1RoiCoor(peakSet)
   outputMatrix=zeros(size(peakSet,1),4);
   for peak=1:size(peakSet,1)

outputMatrix(peak,:)=[peakSet(peak,1),peakSet(peak,1),peakSet(peak,2),peakSet(peak,2)
];

   end


function [outputMatrix] = get2by2RoiCoor(d, peakSet)

      outputMatrix=zeros(size(peakSet,1),4);

      %store bright pixels locations as [x1 x2 y1 y2] or [c1 c2 r1 r2], x1=smalles x and
x2=highest x
%       RoiLength=round(RoiLength);
%       radius=1+ceil(RoiLength/2);

   for peak=1:size(peakSet,1)

      xC=peakSet(peak,1);
      yC=peakSet(peak,2);

      %get a r+1 by r+1 roi
      %get the sum of all possible four r+1 by r+1 squares arround the
      %center of the peak

      sum_1=sum(sum(d(yC-1:yC,xC:xC+1)));
      sum_2=sum(sum(d(yC-1:yC,xC-1:xC)));
      sum_3=sum(sum(d(yC:yC+1,xC-1:xC)));
      sum_4=sum(sum(d(yC:yC+1,xC:xC+1)));
      Max_sum=max([sum_1,sum_2,sum_3,sum_4]);

      tempMatrix=zeros(1,4);
```

```
        if Max_sum==sum_1;
           tempMatrix(1,1)=xC;
           tempMatrix(1,2)=xC+1;
           tempMatrix(1,3)=yC-1;
           tempMatrix(1,4)=yC;

        elseif Max_sum==sum_2;
           tempMatrix(1,1)=xC-1;
           tempMatrix(1,2)=xC;
           tempMatrix(1,3)=yC-1;
           tempMatrix(1,4)=yC;

        elseif Max_sum==sum_3;
           tempMatrix(1,1)=xC-1;
           tempMatrix(1,2)=xC;
           tempMatrix(1,3)=yC;
           tempMatrix(1,4)=yC+1;

        elseif Max_sum==sum_4;
           tempMatrix(1,1)=xC;
           tempMatrix(1,2)=xC+1;
           tempMatrix(1,3)=yC;
           tempMatrix(1,4)=yC+1;
        end
        outputMatrix(peak,:)=tempMatrix;

    end


function [outputMatrix] = get3by3RoiCoor(d, inMat)
        %inMat is the output of get2by2RoiCoor=[x1,x2,y1,y2]

        outputMatrix=zeros(size(inMat,1),4);
        %store bright pixels locations as [x1 x2 y1 y2] or [c1 c2 r1 r2], x1=smalles x and
x2=highest x

        %strategy, make a 3 by 3 square centering each pixels obtained from
        %2by2 matrix


    for peak=1:size(inMat,1)

        x1=inMat(peak,1); x2=inMat(peak,2);
        y1=inMat(peak,3); y2=inMat(peak,4);
```

```
    %make add one half parimeter with one pixel higher length and width centering each
corner of previous roi

    sum_1=sum(sum(d(y1-1:y2,x1-1:x2)));
    sum_2=sum(sum(d(y1-1:y2,x1:x2+1)));

    sum_3=sum(sum(d(y1:y2+1,x1-1:x2)));
    sum_4=sum(sum(d(y1:y2+1,x1:x2+1)));


    Max_sum=max([sum_1,sum_2,sum_3,sum_4]);

    tempMatrix=zeros(1,4);
    if Max_sum==sum_1;
       tempMatrix(1,1)=x1-1;
       tempMatrix(1,2)=x2;
       tempMatrix(1,3)=y1-1;
       tempMatrix(1,4)=y2;

    elseif Max_sum==sum_2;
       tempMatrix(1,1)=x1;
       tempMatrix(1,2)=x2+1;
       tempMatrix(1,3)=y1-1;
       tempMatrix(1,4)=y2;

    elseif Max_sum==sum_3;
       tempMatrix(1,1)=x1-1;
       tempMatrix(1,2)=x2;
       tempMatrix(1,3)=y1;
       tempMatrix(1,4)=y2+1;

    elseif Max_sum==sum_4;
       tempMatrix(1,1)=x1;
       tempMatrix(1,2)=x2+1;
       tempMatrix(1,3)=y1;
       tempMatrix(1,4)=y2+1;
    end
    outputMatrix(peak,:)=tempMatrix;
  end



function [outputMatrix] = get4by4RoiCoor(d, inMat)
  outputMatrix=zeros(size(inMat,1),4);
    %store bright pixels locations as [x1 x2 y1 y2] or [c1 c2 r1 r2], x1=smalles x and
x2=highest x
```

```
    %strategy, add


for peak=1:size(inMat,1)

    x1=inMat(peak,1); x2=inMat(peak,2);
    y1=inMat(peak,3); y2=inMat(peak,4);

    %make add one half parimeter centering each corner of previous roi

    sum_1=sum(sum(d(y1-1:y2,x1-1:x2)));
    sum_2=sum(sum(d(y1-1:y2,x1:x2+1)));
    sum_3=sum(sum(d(y1:y2+1,x1-1:x2)));
    sum_4=sum(sum(d(y1:y2+1,x1:x2+1)));


    Max_sum=max([sum_1,sum_2,sum_3,sum_4]);

    tempMatrix=zeros(1,4);
    if Max_sum==sum_1;
       tempMatrix(1,1)=x1-1;
       tempMatrix(1,2)=x2;
       tempMatrix(1,3)=y1-1;
       tempMatrix(1,4)=y2;

    elseif Max_sum==sum_2;
       tempMatrix(1,1)=x1;
       tempMatrix(1,2)=x2+1;
       tempMatrix(1,3)=y1-1;
       tempMatrix(1,4)=y2;

    elseif Max_sum==sum_3;
       tempMatrix(1,1)=x1-1;
       tempMatrix(1,2)=x2;
       tempMatrix(1,3)=y1;
       tempMatrix(1,4)=y2+1;

    elseif Max_sum==sum_4;
       tempMatrix(1,1)=x1;
       tempMatrix(1,2)=x2+1;
       tempMatrix(1,3)=y1;
       tempMatrix(1,4)=y2+1;
    end
    outputMatrix(peak,:)=tempMatrix;
```

```
    end

function [outputMatrix] = get5by5RoiCoor(d, inMat)
    outputMatrix=zeros(size(inMat,1),4);
        %store bright pixels locations as [x1 x2 y1 y2] or [c1 c2 r1 r2], x1=smalles x and
x2=highest x

        %strategy, add


    for peak=1:size(inMat,1)

        x1=inMat(peak,1); x2=inMat(peak,2);
        y1=inMat(peak,3); y2=inMat(peak,4);

        %make add one half parimeter centering each corner of previous roi

        sum_1=sum(sum(d(y1-1:y2,x1-1:x2)));
        sum_2=sum(sum(d(y1-1:y2,x1:x2+1)));
        sum_3=sum(sum(d(y1:y2+1,x1-1:x2)));
        sum_4=sum(sum(d(y1:y2+1,x1:x2+1)));


        Max_sum=max([sum_1,sum_2,sum_3,sum_4]);

        tempMatrix=zeros(1,4);
        if Max_sum==sum_1;
            tempMatrix(1,1)=x1-1;
            tempMatrix(1,2)=x2;
            tempMatrix(1,3)=y1-1;
            tempMatrix(1,4)=y2;

        elseif Max_sum==sum_2;
            tempMatrix(1,1)=x1;
            tempMatrix(1,2)=x2+1;
            tempMatrix(1,3)=y1-1;
            tempMatrix(1,4)=y2;

        elseif Max_sum==sum_3;
            tempMatrix(1,1)=x1-1;
            tempMatrix(1,2)=x2;
            tempMatrix(1,3)=y1;
            tempMatrix(1,4)=y2+1;

        elseif Max_sum==sum_4;
            tempMatrix(1,1)=x1;
```

```
        tempMatrix(1,2)=x2+1;
        tempMatrix(1,3)=y1;
        tempMatrix(1,4)=y2+1;
    end
    outputMatrix(peak,:)=tempMatrix;

end


function [outputMatrix] = get6by6RoiCoor(d, inMat)
  outputMatrix=zeros(size(inMat,1),4);
    %store bright pixels locations as [x1 x2 y1 y2] or [c1 c2 r1 r2], x1=smalles x and
x2=highest x

    %strategy, add


  for peak=1:size(inMat,1)

    x1=inMat(peak,1); x2=inMat(peak,2);
    y1=inMat(peak,3); y2=inMat(peak,4);

    %make add one half parimeter centering each corner of previous roi

    sum_1=sum(sum(d(y1-1:y2,x1-1:x2)));
    sum_2=sum(sum(d(y1-1:y2,x1:x2+1)));
    sum_3=sum(sum(d(y1:y2+1,x1-1:x2)));
    sum_4=sum(sum(d(y1:y2+1,x1:x2+1)));


    Max_sum=max([sum_1,sum_2,sum_3,sum_4]);

    tempMatrix=zeros(1,4);
    if Max_sum==sum_1;
      tempMatrix(1,1)=x1-1;
      tempMatrix(1,2)=x2;
      tempMatrix(1,3)=y1-1;
      tempMatrix(1,4)=y2;

    elseif Max_sum==sum_2;
      tempMatrix(1,1)=x1;
      tempMatrix(1,2)=x2+1;
      tempMatrix(1,3)=y1-1;
      tempMatrix(1,4)=y2;

    elseif Max_sum==sum_3;
```

```
        tempMatrix(1,1)=x1-1;
        tempMatrix(1,2)=x2;
        tempMatrix(1,3)=y1;
        tempMatrix(1,4)=y2+1;

    elseif Max_sum==sum_4;
        tempMatrix(1,1)=x1;
        tempMatrix(1,2)=x2+1;
        tempMatrix(1,3)=y1;
        tempMatrix(1,4)=y2+1;
    end
    outputMatrix(peak,:)=tempMatrix;

  end

function [backRoiCoordinates] = getBackgndRoiCord(RoiCoordinates,Im)
    backRoiCoordinates=zeros(size(RoiCoordinates));
    for peak=1:size(RoiCoordinates,1)
            bx=2;
            x1= RoiCoordinates(peak,1)-bx; x2=RoiCoordinates(peak,2)+bx;
            y1=RoiCoordinates(peak,3)-bx; y2=RoiCoordinates(peak,4)+bx;
            %if roi dimenstion exceeds the limits

            if x1<1 || y1<1 || x2>size(Im,2)|| y2>size(Im,1)
               if x1<1
                  x1=1;
               end
               if  y1<1
                  y1=1;
               end

                if x2>size(Im,2);
                  x2=size(Im,2);
                end
               if y2>size(Im,1)
                  y2=size(Im,1);
               end
            end

            backRoiCoordinates(peak,:)=[x1, x2, y1, y2];
%               backRoi=rawData(y1:y2,x1:x2,frame);
%             back=[backRoi(:,1)',backRoi(:,end)',backRoi(1,2:end-1),backRoi(end,2:end-
1)];
%               meanBack=mean(back(:));

    end
```

Outline of the GUI: simpleGraph_byFaruk.fig

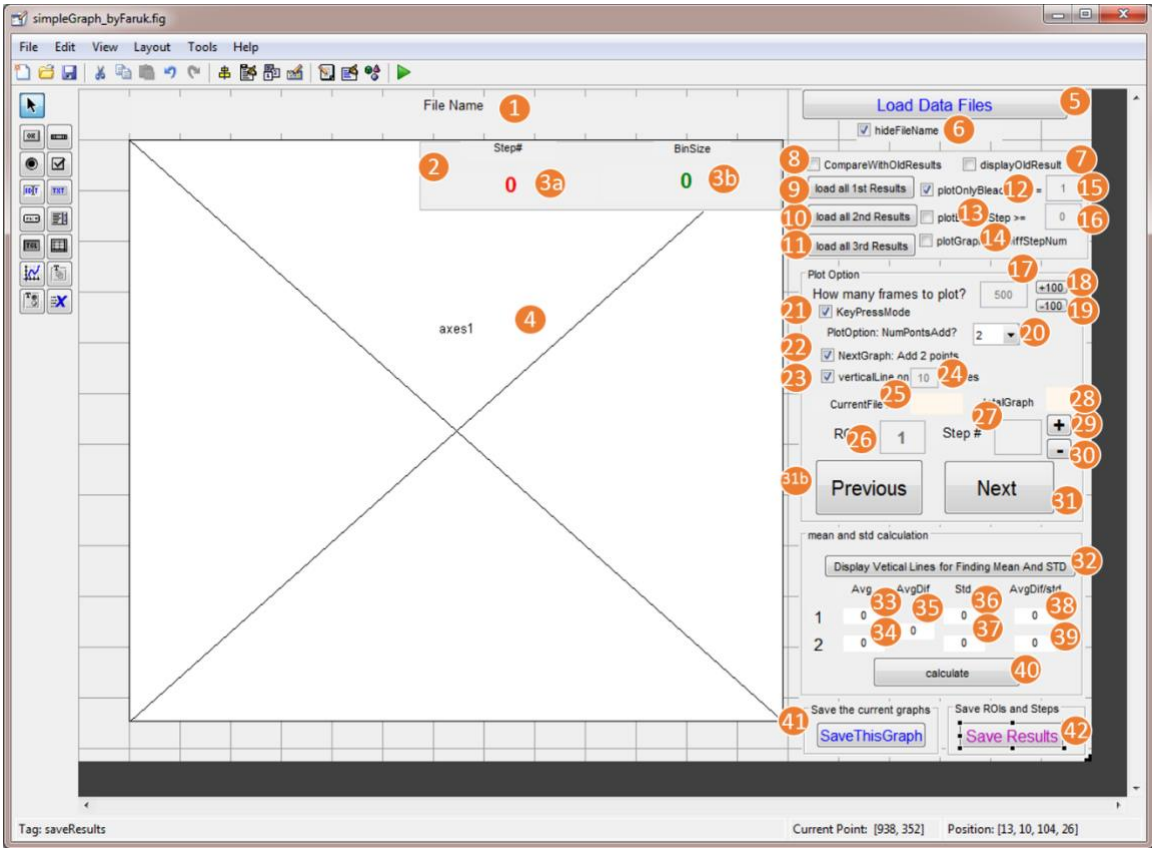

Figure S3: Outline of the GUI used to plot time traces collected using getTimeTraces_byFaruk.fig and getTimeTraces_byFaruk.m. This GUI was saved as "simpleGraph_byFaruk.fig". The underlying variables (Tag) for components 1-42 can be found in the Table S2.

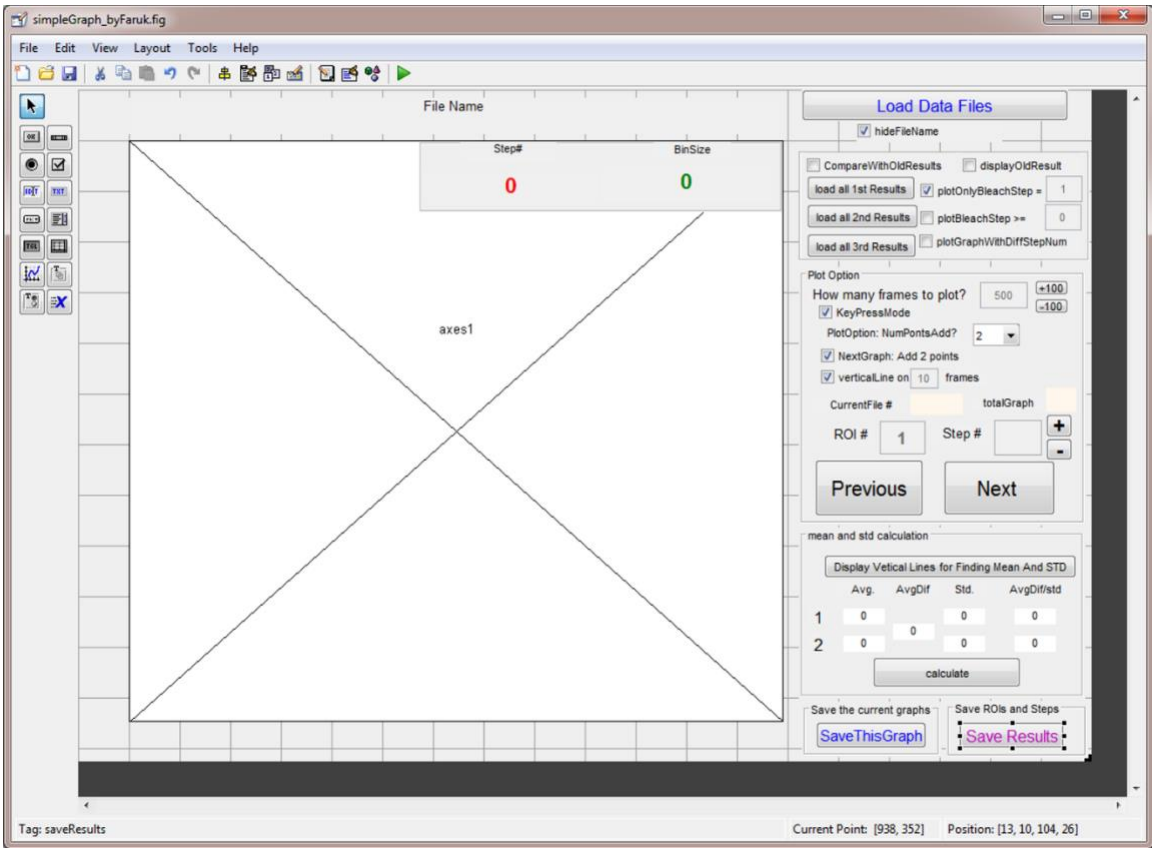

Figure S4: A figure of the GUI shown in Figure S3 without numbering. This figure displays all texts and default values on the GUI.

Table S2: The essential variables used in the figure for GUI (Figure S3). All other components were either Static Texts or Panels.

Number in Figure S1	Type of component	String	Tag
1	Static Text	File Name	fileName
2	Panel	Not applicable	uipanel_DisplayOldResults
3a	Static Text	0 (red)	tb_dispOldStepNum
3b	Static Text	0 (green)	tb_oldBinSize
4	Axes	Not applicable	axes1
5	Push Button	Load Data Files	loadDataFiles
6	Check Box	hideFileName	hideFileName
7	Check Box	displayOldResult	displayOldResult
8	Check Box	CompareWithOldResults	cb_CompareWithOldResults
9	Push Button	load all 1st Results	pb_load_1st_Results
10	Push Button	load all 2nd Results	pb_load_2nd_Results
11	Push Button	load all 3rd Results	pb_load_3rd_Results
12	Check Box	plotOnlyBleachStep =	plotOnlyBleachStepOption
13	Check Box	plotBleachStep >=	plotBleachStepOption_greaterThan
14	Check Box	plotGraphWithDiffStepNum	cb_plotGraphWithDiffStepNum
15	Edit Text	1	plotOnlyBleachStepInput
16	Edit Text	0	plotOnlyBleachStepOption_greaterThanInput
17	Edit Text	500	plotOpt_frameNum

18	Push Button	+100	increaseBy100
19	Push Button	-100	decreaseBy100
20	Pop-up Manu	1;2;3;4;5;6;7;8;9;10 (;=enter)	NumPontsAddToPlot
21	Check Box	keyPressMode	keyPressMode
22	Check Box	NextGraph: Add 2 points	NextGraphPlotAddTwoFrames
23	Check Box	verticalLine on	verticalLine
24	Edit Text	10	verticalLineInput
25	Edit Text	Empty	displayCurrentDataFileNum
26	Edit Text	1	roiNum
27	Edit Text	Empty	stepNum
28	Edit Text	Empty	tb_totalGraph
29	Push Button	+	increaseStepNum
30	Push Button	-	decreaseStepNum
31	Push Button	Next	nextRoi
31b	Push Button	Previous	previousRoi
32	Push Button	Display Vetical Lines for Finding Mean And STD	displayVeticalLinesforMeanAnd STD
33	Edit Text	0	avg_1
34	Edit Text	0	avg_2
35	Edit Text	0	avgDif
36	Edit Text	0	std_1
37	Edit Text	0	std_2
38	Edit Text	0	ratio_1

39	Edit Text	0	ratio_2
40	Edit Text	calculate	calculate
41	Push Button	SaveThisGraph	saveThisGraph
42	Push Button	Save Results	saveResults

Matlab Code in a file called, “simpleGraph_byFaruk.m”

The following code was saved on a Matlab file, called, “simpleGraph_byFaruk.m”.

```
function varargout = simpleGraph_byFaruk(varargin)
% SIMPLEGRAPH_BYFARUK MATLAB code for simpleGraph_byFaruk.fig

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @simpleGraph_byFaruk_OpeningFcn, ...
                   'gui_OutputFcn',  @simpleGraph_byFaruk_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before simpleGraph_byFaruk is made visible.
function simpleGraph_byFaruk_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to simpleGraph_byFaruk (see VARARGIN)

% Choose default command line output for simpleGraph_byFaruk
   handles.output = hObject;
%update some presets

    set(hObject,                                                    'WindowButtonUpFcn',
@(hObject,eventdata)simpleGraph_byFaruk('stopDragFcn',hObject,eventdata,guidata(hO
bject)));
   set(handles.axes1,'XTickLabel','','YTickLabel','','xtick',[],'ytick',[]);
%     set(handles.axes2,'XTickLabel','','YTickLabel','','xtick',[],'ytick',[]);
```

```
%     set(handles.plotOpt_frameNum,'String','1000');
%     set(handles.cb_CompareWithOldResults,'Value',0);
%     set(handles.cb_CompareWithOldResults,'Visible','off');
%        set(handles.uipanel_DisplayOldResults,'Visible','off');
%     set(handles.pb_load_2nd_Results,'Visible','off');
%     set(handles.pb_load_3rd_Results,'Visible','off');
%     set(handles.cb_plotGraphWithDiffStepNum,'Visible','off');
%     set(handles.cb_plotGraphWithDiffStepNum,'Value',0)
%     set(handles.plotBleachStepOption_greaterThan,'Visible','off');
%     set(handles.plotBleachStepOption_greaterThan,'Value',0);
%
% %
% %     set(get(handles.uipanel_findMeanAndStd,'Children'), 'Enable','off');
% %     set(handles.plotOpt_frameNum,'Enable','off');
% %     set(handles.roiNum,'Enable','off');
% %    set(handles.stepNum,'Enable','off');
% %     set(handles.verticalLineInput,'Enable','off');
% %     set(handles.plotOnlyBleachStepInput,'Enable','off');
% %     set(handles.plotOnlyBleachStepOption_greaterThanInput,'Enable','off');


%     set(get(handles.uipanel_plotOpt_Limits,'Children'), 'Enable','off');

%set the name of GUI
    set(handles.figure1,'Name',mfilename);

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes simpleGraph_byFaruk wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = simpleGraph_byFaruk_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

function figure1_WindowKeyPressFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  structure with the following fields (see FIGURE)
```

```
%        Key: name of the key that was pressed, in lower case
%        Character: character interpretation of the key(s) that was pressed
%        Modifier: name(s) of the modifier key(s) (i.e., control, shift) pressed
% handles    structure with handles and user data (see GUIDATA)

%keypressed=get(handles.figure1,'CurrentCharacter');
keypressed=eventdata.Character;
% Determine whether Enter was pressed to request new crosshairs
keypressMode=get(handles.keyPressMode,'Value');

if keypressMode==1;

switch keypressed

   % for next images
%     case 119 %119 is the ascii value for w
%       nextRoi_Callback(hObject, eventdata, handles)

   case 30 %30 is the ascii value for upwards arrow
     nextRoi_Callback(hObject, eventdata, handles)

   case 29 %29 is the ascii value for rightwards arrow
     nextRoi_Callback(hObject, eventdata, handles)

   % for previous images
%     case 100 %100 is the ascii value for w
%      previousRoi_Callback(hObject, eventdata, handles)
%
   case 31 %31 is the ascii value for downwards arrow
    previousRoi_Callback(hObject, eventdata, handles)

   case 28 %28 is the ascii value for leftwards arrow
   previousRoi_Callback(hObject, eventdata, handles)

    %number in inputs

    case 13 %press enter
    noSteps_Callback(hObject, eventdata, handles)
    nextRoi_Callback(hObject, eventdata, handles)

    case 101 %press e
    noSteps_Callback(hObject, eventdata, handles)
    nextRoi_Callback(hObject, eventdata, handles)

    case 48
    zero_Callback(hObject, eventdata, handles)
```

```
 nextRoi_Callback(hObject, eventdata, handles)

 case 96 %key `
 zero_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)

case 49
 one_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)

case 50
 two_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)

 case 51
 three_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)
 case 52
 four_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)
 case 53
 five_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)
 case 54
 six_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)
 case 55
 seven_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)
 case 56
 eight_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)
 case 57
 nine_Callback(hObject, eventdata, handles)
 nextRoi_Callback(hObject, eventdata, handles)

 case 43 %+ key
    if get(handles.NumPontsAddToPlot,'Value')==1
       set(handles.NumPontsAddToPlot,'Value',2)
    else
        set(handles.NumPontsAddToPlot,'Value',1)
    end
    handles=plotSimpleGraph_newApproach(handles) ;


 case 45 % - key
```

```
        if get(handles.NumPontsAddToPlot,'Value')==1
           set(handles.NumPontsAddToPlot,'Value',2)
        else
            set(handles.NumPontsAddToPlot,'Value',1)
        end
        handles=plotSimpleGraph_newApproach(handles) ;
end

end


%  --- Executes on button press in loadDataFiles.
function loadDataFiles_Callback(hObject, eventdata, handles)
% hObject    handle to loadDataFiles (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


   pathMfile = fileparts(mfilename('fullpath'));
   addpath(pathMfile);
%        addpath(fullfile(pathMfile, 'simpleGraph_calls'));

   %addpath([pathMfile, '\vbFRET']);
   %open a dialoge box to get a file
   [fileNames,pathName]=uigetfile('*.xlsx','Please select all the files you want to
analyze','MultiSelect','on');


   %check if user did not select a file
   if iscell(fileNames)
     handles. fileNames=fileNames;
   elseif fileNames==0
      return
   else
      afile={};
      afile{1}=fileNames;
      handles.fileNames=afile;
   end
   handles.curFileNum=1;

set(handles.displayCurrentDataFileNum,'String',[num2str(handles.curFileNum),'/',num2st
r(length(handles.fileNames))]);
   handles.pathName=pathName;
   handles=
LoadADataFile(handles,handles.fileNames{handles.curFileNum},handles.pathName);
```

```
      set(handles.cb_CompareWithOldResults,'Visible','on','Value',0);
%       set(handles.displayOldResult,'Visible','on','Value',0);
       set(handles.pb_load_1st_Results,'Visible','off');
       set(handles.pb_load_2nd_Results,'Visible','off');
       set(handles.pb_load_3rd_Results,'Visible','off');
       set(handles.cb_plotGraphWithDiffStepNum,'Visible','off');
%        set(handles.cb_plotGraphWithDiffStepNum,'Value',0)
       set(handles.plotBleachStepOption_greaterThan,'Visible','off');
%        set(handles.plotBleachStepOption_greaterThan,'Value',0)
       set(handles.plotOnlyBleachStepOption,'Visible','off');
       set(handles.plotOnlyBleachStepInput,'Visible','off');
          set(handles.tb_dispOldStepNum,'String','');
       set(handles.tb_oldBinSize,'String','');

       %for calculating mean and std
       handles.calculate_1Data=[];
       handles.calculate_2Data=[];
       handles.calculate_3Data=[];
       set(handles.NumPontsAddToPlot,'Value',2)

guidata(hObject, handles)


% --- Executes on button press in saveThisGraph.
function saveThisGraph_Callback(hObject, eventdata, handles)
% hObject    handle to saveThisGraph (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%    structure with handles and user data (see GUIDATA)
h = waitbar(0,'Please wait...'); %open a waitbar so that user can see that matlab has done
something upon pressin this button


currentRoi=get(handles.roiNum,'String'); %get the graph number which is actually ROI
number
stepNum=get(handles.stepNum,'String'); %get the number number
path=handles.datafilePathname; %same as filename
fileName=handles.fileNames{handles.curFileNum};

if strcmp(fileName(end-3:end),'xlsx')==1;
   fileName=fileName(1:end-5);%remove .xlsx from the file name
else
   fileName=fileName(1:end-4);
end
% concatinate ROI number with ROIs and step number with Step to add these
```

```
% into the final file name of the graph
ROI=['- ROI- ',currentRoi];
step=[' Step-',stepNum];

fileName=strcat(fileName,ROI,step); %concatinate all to make final file name
fileName=['Data of-',fileName,'.xlsx'];
dataFileName=fullfile(path,fileName);%add path infront of file name
figHand = findobj(handles.axes1,'Type','line');

plottedData=num2cell(handles.plottedData);
headings=['Num of points added', num2str(get(handles.NumPontsAddToPlot,'Value'))];

plottedData=[headings;plottedData];


%
% %if user did not check the checkbox 'plot residual data', then we will not
% %have last two label on the plottedDataHeading
% plotOpt_residualData=get(handles.plotOpt_residualData,'Value');
% if plotOpt_residualData==0
%     dataHeading=handles.plottedDataHeaing(1:end-2);
% else
%     dataHeading=handles.plottedDataHeaing;
% end
% plottedData=vertcat(dataHeading, plottedData);
if ismac()
   my_xlwrite(dataFileName,plottedData,'sheet', 'A1')
else
   xlswrite(dataFileName,plottedData);
end

% %save the graph
% F=getframe(handles.axes1); %Capture the figuref from GUI as movie frame
 close(h) ;% closing my waitbar as I can not close it after this lines :(
% h=figure(); %new figure
% image(F.cdata); %F is a structure with the frame and cdata contains the captured image
data. and make a new image with that cdata
h=figure();
% plotSimpleGraph(gca,handles.plottedData)
plot(handles.plottedData)
saveas(gca, [fileName(1:end-5),'.fig'], 'fig'); %save current figure handles as assinged file
name and format
close(h); %and close it


% --- Executes on button press in saveResults.
```

```
function saveResults_Callback(hObject, eventdata, handles)
% hObject    handle to saveResults (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
h=waitbar(0,'Please wait! saving the result');
outputData=handles.outputData;

path=handles.datafilePathname; %same as filename

fileName=handles.fileNames{handles.curFileNum};

if strcmp(fileName(end-3:end),'xlsx')==1;
   fileName=fileName(1:end-5);%remove .xlsx from the file name
else
   fileName=fileName(1:end-4);
end
%make a new file name adding a part infront of it

fileAddfront='Result of- ';
fileAtEnd='.txt';
fileName=strcat(fileAddfront,fileName,fileAtEnd);

%mane the file name with path in it

fileName=fullfile(path,fileName);
if exist(fileName,'file')==2
       delete(fileName);
end
% xlswrite(fileName,num2cell(outputData))


   fileID = fopen(fileName, 'wt');
   fprintf(fileID,'ROIs #\tStep #\tBinSize\txAxisSize\n'); %add heading in each column as
ROI and step #
   % fprintf(fileID,'%g\t%g\n',outputData.');
   fprintf(fileID,'%g\t%g\t%g\t%g\n',outputData.');
   fclose(fileID);
% end
close(h) ;


% --- Executes on button press in nextRoi.
function nextRoi_Callback(hObject, eventdata, handles)
% hObject    handle to nextRoi (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% set(handles.CKFiltering,'Value',0);
%with currect ROI
currentRoi=str2double(get(handles.roiNum,'String'));
   %record results
   binSize=get(handles.NumPontsAddToPlot,'Value');
   lim = axis;
   xLim=lim(2);
   %under key press fnc, input this value.
         currentStep=str2double(get(handles.stepNum,'String'));
         handles.outputData(currentRoi,:)=[currentRoi,currentStep,binSize,xLim];

   %plot next graph
   nextRoi=currentRoi+1;

      %for comparing with old results and automatically plot next graph
      %with different stepNum,
       if get(handles.plotOnlyBleachStepOption,'Value')
          aVal=str2double(get(handles.plotOnlyBleachStepInput,'String'));
          if isfield(handles,'old_3rd_Results') && ~isempty(handles.old_3rd_Results);

nextRoi=getNextRoi_aValue(aVal,currentRoi,handles.old_1st_Results,handles.old_2nd_
Results,handles.old_3rd_Results);
             if nextRoi-currentRoi>1; %next roi is not selected
             handles.outputData(currentRoi+1:nextRoi-
1,:)=handles.old_3rd_Results(currentRoi+1:nextRoi-1,:);
             end
          elseif isfield(handles,'old_2nd_Results') && ~isempty(handles.old_2nd_Results);

nextRoi=getNextRoi_aValue(aVal,currentRoi,handles.old_1st_Results,handles.old_2nd_
Results);
              if nextRoi-currentRoi>1; %next roi is not selected
              handles.outputData(currentRoi+1:nextRoi-
1,:)=handles.old_2nd_Results(currentRoi+1:nextRoi-1,:);
              end
          elseif isfield(handles,'old_1st_Results') && ~isempty(handles.old_1st_Results);
             nextRoi=getNextRoi_aValue(aVal,currentRoi,handles.old_1st_Results);
             if nextRoi-currentRoi>1; %next roi is not selected
             handles.outputData(currentRoi+1:nextRoi-
1,:)=handles.old_1st_Results(currentRoi+1:nextRoi-1,:);
             end
          end

       elseif get(handles.cb_plotGraphWithDiffStepNum,'Value')
```

```
            if isfield(handles,'old_3rd_Results') && ~isempty(handles.old_3rd_Results);

nextRoi=getNextRoi_differntStepNum(currentRoi,handles.old_1st_Results,handles.old_
2nd_Results,handles.old_3rd_Results);
                if nextRoi-currentRoi>1; %next roi is not selected
                handles.outputData(currentRoi+1:nextRoi-
1,:)=handles.old_3rd_Results(currentRoi+1:nextRoi-1,:);
                end
            elseif isfield(handles,'old_2nd_Results') && ~isempty(handles.old_2nd_Results);

nextRoi=getNextRoi_differntStepNum(currentRoi,handles.old_1st_Results,handles.old_
2nd_Results);
                if nextRoi-currentRoi>1; %next roi is not selected
                handles.outputData(currentRoi+1:nextRoi-
1,:)=handles.old_2nd_Results(currentRoi+1:nextRoi-1,:);
                end
            elseif isfield(handles,'old_1st_Results') && ~isempty(handles.old_1st_Results);
                nextRoi=getNextRoi_differntStepNum(currentRoi,handles.old_1st_Results);
                if nextRoi-currentRoi>1; %next roi is not selected
                handles.outputData(currentRoi+1:nextRoi-
1,:)=handles.old_1st_Results(currentRoi+1:nextRoi-1,:);
                end
            end
        elseif get(handles.plotBleachStepOption_greaterThan,'Value')

aVal=str2double(get(handles.plotOnlyBleachStepOption_greaterThanInput,'String'));
            if isfield(handles,'old_3rd_Results') && ~isempty(handles.old_3rd_Results);

nextRoi=getNextRoi_greaterThan(aVal,currentRoi,handles.old_1st_Results,handles.old_
2nd_Results,handles.old_3rd_Results);
                if nextRoi-currentRoi>1; %next roi is not selected
                handles.outputData(currentRoi+1:nextRoi-
1,:)=handles.old_3rd_Results(currentRoi+1:nextRoi-1,:);
                end
            elseif isfield(handles,'old_2nd_Results') && ~isempty(handles.old_2nd_Results);

nextRoi=getNextRoi_greaterThan(aVal,currentRoi,handles.old_1st_Results,handles.old_
2nd_Results);
                if nextRoi-currentRoi>1; %next roi is not selected
                handles.outputData(currentRoi+1:nextRoi-
1,:)=handles.old_2nd_Results(currentRoi+1:nextRoi-1,:);
                end
            elseif isfield(handles,'old_1st_Results') && ~isempty(handles.old_1st_Results);
                nextRoi=getNextRoi_greaterThan(aVal,currentRoi,handles.old_1st_Results);
                if nextRoi-currentRoi>1; %next roi is not selected
```

```
            handles.outputData(currentRoi+1:nextRoi-
1,:)=handles.old_1st_Results(currentRoi+1:nextRoi-1,:);
            end
          end

      end

   %plot next graph

if nextRoi<=size(handles.rawData,2) %for all the ROIs
   set(handles.roiNum,'String',num2str(nextRoi));
   if get(handles.NextGraphPlotAddTwoFrames,'Value')
      set(handles.NumPontsAddToPlot,'Value',2);
   end


   %plot next graph
   handles=plotSimpleGraph_newApproach(handles) ;


   if  get(handles.cb_CompareWithOldResults,'Value')
      if isfield (handles,'old_3rd_Results') && ~isempty(handles.old_3rd_Results);
         handles=plot_3rdResult(handles); %will plot 2nd and 1st as well
      elseif isfield (handles,'old_2nd_Results') && ~isempty(handles.old_2nd_Results);
         handles=plot_2ndResult(handles); %will pot 1st as well
      elseif isfield (handles,'old_1st_Results') && ~isempty(handles.old_1st_Results);
         handles=plot_1stResult(handles);
      end
   end

else
   %for previous graphNum share maxRoi
   handles.totalRoisLastDataFile=nextRoi-1;
   handles.lastDataOutputData=handles.outputData;
   %save it
   saveResults_Callback(hObject, eventdata, handles);

   %plot next files
   if handles.curFileNum<length(handles.fileNames)
   handles.curFileNum= handles.curFileNum+1;

   handles=
LoadADataFile(handles,handles.fileNames{handles.curFileNum},handles.pathName);

set(handles.displayCurrentDataFileNum,'String',[num2str(handles.curFileNum),'/',num2st
r(length(handles.fileNames))]);
```

```
   handles=load_1stResultFile(handles);
   handles=load_2ndResultFile(handles);
   handles=load_3rdResultFile(handles);
   else
      msgbox([{'All selected files have been analyzed!!!'};{'Its time to get a Beer!!!'}]);
   end

end

  %for calculating mean and std
       handles.calculate_1Data=[];
       handles.calculate_2Data=[];
       handles.calculate_3Data=[];

guidata(hObject,handles);

    % --- Executes on button press in previousRoi.
function previousRoi_Callback(hObject, eventdata, handles)
% hObject    handle to previousRoi (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%         set(handles.CKFiltering,'Value',0);
       %with currect ROI
       currentRoi=str2double(get(handles.roiNum,'String'));

       %for 1st ROI you cannot go to previous
if currentRoi>1

       %for prvious ROI
       previousRoi=currentRoi-1;
       %for comparing with old results and automatically plot next graph
       %with different stepNum,

      if get(handles.plotOnlyBleachStepOption,'Value')
           aVal=str2double(get(handles.plotOnlyBleachStepInput,'String'));
           if isfield(handles,'old_3rd_Results') && ~isempty(handles.old_3rd_Results);

previousRoi=getPreviousRoi_aValue(aVal,currentRoi,handles.old_1st_Results,handles.ol
d_2nd_Results,handles.old_3rd_Results);
          elseif isfield(handles,'old_2nd_Results') && ~isempty(handles.old_2nd_Results);

previousRoi=getPreviousRoi_aValue(aVal,currentRoi,handles.old_1st_Results,handles.ol
d_2nd_Results);
          elseif isfield(handles,'old_1st_Results') && ~isempty(handles.old_1st_Results);

previousRoi=getPreviousRoi_aValue(aVal,currentRoi,handles.old_1st_Results);
```

```
            end
        elseif get(handles.cb_plotGraphWithDiffStepNum,'Value')
            if isfield(handles,'old_3rd_Results') && ~isempty(handles.old_3rd_Results);

previousRoi=getPreviousRoi_differntStepNum(currentRoi,handles.old_1st_Results,handl
es.old_2nd_Results,handles.old_3rd_Results);
            elseif isfield(handles,'old_2nd_Results') && ~isempty(handles.old_2nd_Results);

previousRoi=getPreviousRoi_differntStepNum(currentRoi,handles.old_1st_Results,handl
es.old_2nd_Results);
            elseif isfield(handles,'old_1st_Results') && ~isempty(handles.old_1st_Results);

previousRoi=getPreviousRoi_differntStepNum(currentRoi,handles.old_1st_Results);
            end
        elseif get(handles.plotBleachStepOption_greaterThan,'Value')

aVal=str2double(get(handles.plotOnlyBleachStepOption_greaterThanInput,'String'));
            if isfield(handles,'old_3rd_Results') && ~isempty(handles.old_3rd_Results);

previousRoi=getPreviousRoi_greaterThan(aVal,currentRoi,handles.old_1st_Results,hand
les.old_2nd_Results,handles.old_3rd_Results);
            elseif isfield(handles,'old_2nd_Results') && ~isempty(handles.old_2nd_Results);

previousRoi=getPreviousRoi_greaterThan(aVal,currentRoi,handles.old_1st_Results,hand
les.old_2nd_Results);
            elseif isfield(handles,'old_1st_Results') && ~isempty(handles.old_1st_Results);

previousRoi=getPreviousRoi_greaterThan(aVal,currentRoi,handles.old_1st_Results);
            end
        end


        set(handles.roiNum,'String',num2str(previousRoi));

    %plot previous graph
    if get(handles.NextGraphPlotAddTwoFrames,'Value')
        set(handles.NumPontsAddToPlot,'Value',2);
    end


    %plot next graph
    handles=plotSimpleGraph_newApproach(handles) ;

        %show old results if exist
    if  get(handles.cb_CompareWithOldResults,'Value')
        if isfield (handles,'old_3rd_Results') && ~isempty(handles.old_3rd_Results);
```

```
        handles=plot_3rdResult(handles);
      elseif isfield (handles,'old_2nd_Results') && ~isempty(handles.old_2nd_Results);
        handles=plot_2ndResult(handles);
      elseif isfield (handles,'old_1st_Results') && ~isempty(handles.old_1st_Results);
        handles=plot_1stResult(handles);
      end
   end
elseif currentRoi==1 && handles.curFileNum>1
   uiwait(msgbox('Sorry!! This version does not allow to go prvious movie data'));

%     handles.curFileNum= handles.curFileNum-1;
%
set(handles.displayCurrentDataFileNum,'String',[num2str(handles.curFileNum),'/',num2st
r(length(handles.fileNames))]);
%                                                                                 handles=
LoadADataFile(handles,handles.fileNames{handles.curFileNum},handles.pathName,han
dles.totalRoisLastDataFile);
%     handles=load_1stResultFile(handles);
%     handles=load_2ndResultFile(handles);
%     handles=load_3rdResultFile(handles);
end

 guidata(hObject,handles);

function roiNum_Callback(hObject, eventdata, handles)
% hObject    handle to roiNum (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of roiNum as text
%        str2double(get(hObject,'String')) returns contents of roiNum as a double
handles=plotSimpleGraph_newApproach(handles) ;
  guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function roiNum_CreateFcn(hObject, eventdata, handles)
% hObject    handle to roiNum (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if                ispc                &&                isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
```

```
end

function stepNum_Callback(hObject, eventdata, handles)
% hObject    handle to stepNum (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of stepNum as text
%        str2double(get(hObject,'String')) returns contents of stepNum as a double

% --- Executes during object creation, after setting all properties.
function stepNum_CreateFcn(hObject, eventdata, handles)
% hObject    handle to stepNum (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if                ispc                &&                isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in keyPressMode.
function keyPressMode_Callback(hObject, eventdata, handles)
% hObject    handle to keyPressMode (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of keyPressMode

if get(hObject,'Value')
set(handles.plotOnlyBleachStepInput,'Enable','off');
set(handles.plotOnlyBleachStepOption_greaterThanInput,'Enable','off');
   set(handles.plotOpt_frameNum,'Enable','off');
   set(handles.verticalLineInput,'Enable','off');
   set(handles.roiNum,'Enable','off');
   set(handles.stepNum,'Enable','off');
else
   set(handles.plotOnlyBleachStepInput,'Enable','on');
set(handles.plotOnlyBleachStepOption_greaterThanInput,'Enable','on');
   set(handles.plotOpt_frameNum,'Enable','on');
   set(handles.verticalLineInput,'Enable','on');
   set(handles.roiNum,'Enable','on');
```

```
   set(handles.stepNum,'Enable','on');
end
   guidata(hObject, handles);

function plotOpt_frameNum_Callback(hObject, eventdata, handles)
% hObject    handle to plotOpt_frameNum (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of plotOpt_frameNum as text
%        str2double(get(hObject,'String')) returns contents of plotOpt_frameNum as a double
input=str2double(get(handles.plotOpt_frameNum,'String'));
if input<50
   set(hObject,'String','50')
end
handles=plotSimpleGraph_newApproach(handles) ;
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function plotOpt_frameNum_CreateFcn(hObject, eventdata, handles)
% hObject    handle to plotOpt_frameNum (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if                ispc                &&                isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in increaseBy100.
function increaseBy100_Callback(hObject, eventdata, handles)
% hObject    handle to increaseBy100 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
input=str2double(get(handles.plotOpt_frameNum,'String'));
set(handles.plotOpt_frameNum,'String',num2str(input+100));
handles=plotSimpleGraph_newApproach(handles) ;
guidata(hObject,handles);

% --- Executes on button press in decreaseBy100.
function decreaseBy100_Callback(hObject, eventdata, handles)
% hObject    handle to decreaseBy100 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
input=str2double(get(handles.plotOpt_frameNum,'String'));
if input-100>50
   set(handles.plotOpt_frameNum,'String',num2str(input-100));
else
   set(handles.plotOpt_frameNum,'String','50');
end
handles=plotSimpleGraph_newApproach(handles) ;
guidata(hObject,handles);


% --- Executes on button press in increaseStepNum.
function increaseStepNum_Callback(hObject, eventdata, handles)
% hObject    handle to increaseStepNum (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
stepNum=str2double(get(handles.stepNum,'String'));
val=stepNum+1;
if isnan(stepNum);
   set(handles.stepNum,'String','0');
else
   set(handles.stepNum,'String',num2str(val));
end
guidata(hObject, handles);


% --- Executes on button press in decreaseStepNum.
function decreaseStepNum_Callback(hObject, eventdata, handles)
% hObject    handle to decreaseStepNum (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
stepNum=str2double(get(handles.stepNum,'String'));
val=stepNum-1;
if stepNum==0
    set(handles.stepNum,'String','');%empty
elseif stepNum>0
   set(handles.stepNum,'String',num2str(val));
end
guidata(hObject, handles);


% --- Executes on button press in cb_CompareWithOldResults.
function cb_CompareWithOldResults_Callback(hObject, eventdata, handles)
% hObject    handle to cb_CompareWithOldResults (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hint: get(hObject,'Value') returns toggle state of cb_CompareWithOldResults

   if get(hObject,'Value');
   set(handles.pb_load_1st_Results,'Visible','on');
   set(handles.plotBleachStepOption_greaterThan,'Visible','on');
   set(handles.plotOnlyBleachStepOption,'Visible','on');
   set(handles.plotOnlyBleachStepInput,'Visible','on');
   set(handles.plotOnlyBleachStepOption_greaterThanInput,'Visible','on');
   set(handles.displayOldResult,'Visible','on','Value',0);

   else
   set(handles.pb_load_1st_Results,'Visible','off');
   set(handles.pb_load_2nd_Results,'Visible','off');
   set(handles.pb_load_3rd_Results,'Visible','off');
   set(handles.plotBleachStepOption_greaterThan,'Visible','off','Value',0);
   set(handles.plotOnlyBleachStepOption,'Visible','off');
   set(handles.plotOnlyBleachStepInput,'Visible','off');
   set(handles.plotOnlyBleachStepOption_greaterThanInput,'Visible','off');
   set(handles.displayOldResult,'Visible','off','Value',0);
   end
%     handles= load_1stResultFile(handles);
if get(handles.displayOldResult,'Value')
   set(handles.uipanel_DisplayOldResults,'Visible','on');
else
   set(handles.uipanel_DisplayOldResults,'Visible','off');
end
guidata(hObject, handles);

% function handles=load_anyResultsFile(handles, resultFileName,PathName)


    %display off ylabel
% --- Executes on button press in pb_load_1st_Results.
function pb_load_1st_Results_Callback(hObject, eventdata, handles)
% hObject    handle to pb_load_1st_Results (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

set(handles.pb_load_2nd_Results,'Visible','on');
[resultFileNames,pathName]=uigetfile('*.txt','Please  select  all  the  files  you  want  to
analyze','MultiSelect','on');


   %check if user did not select a file
   if iscell(resultFileNames)
```

```
      handles. old_1st_resultFileNames=resultFileNames;
   elseif resultFileNames==0
      return
   else
      afile={};
      afile{1}=resultFileNames;
      handles.old_1st_resultFileNames=afile;
   end

   handles.old_1st_resultpathName=pathName;
   handles=load_1stResultFile(handles);

guidata(hObject, handles);


function handles=load_1stResultFile(handles)

   if ~isfield(handles,'old_1st_resultpathName');
      return;
   end

    curDir=cd;
   if isfield(handles,'path_load_1nd_Results')
      cd(handles.old_1st_resultpathName);
   end

   %change the directory again
   cd(curDir)

   pathName=handles.old_1st_resultpathName;
   curDataFileName=handles.fileNames{handles.curFileNum};

resultFileNameCorspndToDataFileName=getCorFileName(curDataFileName,handles.old
_1st_resultFileNames);
   if ~isempty(resultFileNameCorspndToDataFileName)
      fileName=resultFileNameCorspndToDataFileName;
      fullFileName=fullfile(pathName,fileName);
      handles.path_CompareWithOldResults=pathName;
      [data,~]=importdata(fullFileName);
      oldResults=data.data;
      handles.old_1st_Results=oldResults;
   else
      handles.old_1st_Results=[];
   end
   handles=plot_1stResult(handles);
```

```
% --- Executes on button press in pb_load_2nd_Results.
function pb_load_2nd_Results_Callback(hObject, eventdata, handles)
% hObject    handle to pb_load_2nd_Results (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
 set(handles.pb_load_3rd_Results,'Visible','on');
[resultFileNames,pathName]=uigetfile('*.txt','Please select all the files you want to analyze','MultiSelect','on');


    %check if user did not select a file
    if iscell(resultFileNames)
      handles. old_2nd_resultFileNames=resultFileNames;
    elseif resultFileNames==0
        return
    else
        afile={};
        afile{1}=resultFileNames;
        handles.old_2nd_resultFileNames=afile;
    end

    handles.old_2nd_resultpathName=pathName;
    handles=load_2ndResultFile(handles);
    guidata(hObject, handles);

    function handles=load_2ndResultFile(handles)
    if ~isfield(handles,'old_2nd_resultpathName');
        return;
    end

    curDir=cd;
    if isfield(handles,'path_load_2nd_Results')
        cd(handles.old_2nd_resultpathName);
    end

    %change the directory again
    cd(curDir);

    pathName=handles.old_2nd_resultpathName;
    curDataFileName=handles.fileNames{handles.curFileNum};

resultFileNameCorspndToDataFileName=getCorFileName(curDataFileName,handles.old_2nd_resultFileNames);
    if ~isempty(resultFileNameCorspndToDataFileName)
    fileName=resultFileNameCorspndToDataFileName;
          fullFileName=fullfile(pathName,fileName);
```

```
    [data,~]=importdata(fullFileName);
    oldResults=data.data;
    handles.old_2nd_Results=oldResults;
    else
          handles.old_2nd_Results=[];
    end
    handles=plot_2ndResult(handles);

    set(handles.cb_plotGraphWithDiffStepNum,'Visible','on');
% --- Executes on button press in pb_load_3rd_Results.
function pb_load_3rd_Results_Callback(hObject, eventdata, handles)
% hObject    handle to pb_load_3rd_Results (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
  [resultFileNames,pathName]=uigetfile('*.txt','Please select all the files you want to
analyze','MultiSelect','on');


    %check if user did not select a file
    if iscell(resultFileNames)
      handles. old_3rd_resultFileNames=resultFileNames;
    elseif resultFileNames==0
       return
    else
       afile={};
       afile{1}=resultFileNames;
       handles.old_3rd_resultFileNames=afile;
    end

    handles.old_3rd_resultpathName=pathName;
    handles=load_3rdResultFile(handles)  ;
guidata(hObject, handles);

function handles=load_3rdResultFile(handles)
    if ~isfield(handles,'old_3rd_resultpathName');
       return;
    end

    curDir=cd;
    if isfield(handles,'path_load_3rd_Results')
       cd(handles.old_3rd_resultpathName);
    end

    %change the directory again
    cd(curDir);
```

```
    pathName=handles.old_3rd_resultpathName;
    curDataFileName=handles.fileNames{handles.curFileNum};

resultFileNameCorspndToDataFileName=getCorFileName(curDataFileName,handles.old
_3rd_resultFileNames);
    if ~isempty(resultFileNameCorspndToDataFileName)
        fileName=resultFileNameCorspndToDataFileName;
        fullFileName=fullfile(pathName,fileName);
        [data,~]=importdata(fullFileName);
        oldResults=data.data;
        handles.old_3rd_Results=oldResults;
    else
         handles.old_3rd_Results=[];
    end
    handles=plot_3rdResult(handles);

   function handles=plot_1stResult(handles)
    if get(handles.displayOldResult,'Value')==0
      return
    end
%for displaying old results
    if ~isempty(handles.roiNum)
        currentRoi=str2double(get(handles.roiNum,'String'));
    else
        currentRoi=1;
    end
            oldStepNum=num2str(handles.old_1st_Results(currentRoi,2));
            set(handles.tb_dispOldStepNum,'String',oldStepNum);

%         set(handles.cb_plotGraphWithDiffStepNum,'Value',1);
%         set(handles.plotBleachStepOption_greaterThan,'Value',0);
            if size(handles.old_1st_Results,2)>2 %not previous version's analysis data
            oldBin=num2str(handles.old_1st_Results(currentRoi,3));
        set(handles.tb_oldBinSize,'String',oldBin);
    end

 function handles=plot_2ndResult(handles)
    if get(handles.displayOldResult,'Value')==0
      return
    end
%for displaying old results
    if ~isempty(handles.roiNum)
        currentRoi=str2double(get(handles.roiNum,'String'));
    else
        currentRoi=1;
    end
```

```
        oldStepNum=[num2str(handles.old_1st_Results(currentRoi,2)),'-',...
           num2str(handles.old_2nd_Results(currentRoi,2))];
        set(handles.tb_dispOldStepNum,'String',oldStepNum);
%
%       set(handles.cb_plotGraphWithDiffStepNum,'Value',1);
%       set(handles.plotBleachStepOption_greaterThan,'Value',0);
        if size(handles.old_1st_Results,2)>2 %not previous version's analysis data
        oldBin=[num2str(handles.old_1st_Results(currentRoi,3)),'-',...
                num2str(handles.old_2nd_Results(currentRoi,3))];
        set(handles.tb_oldBinSize,'String',oldBin);

   end

%for displaying old results



   function handles=plot_3rdResult(handles)

 if get(handles.displayOldResult,'Value')==0
    return
 end
%for displaying old results
   if ~isempty(handles.roiNum)
      currentRoi=str2double(get(handles.roiNum,'String'));
   else
      currentRoi=1;
   end


        oldStepNum=[num2str(handles.old_1st_Results(currentRoi,2)),'-',...
           num2str(handles.old_2nd_Results(currentRoi,2)),'-',...
           num2str(handles.old_3rd_Results(currentRoi,2))];
        set(handles.tb_dispOldStepNum,'String',oldStepNum);

%       set(handles.cb_plotGraphWithDiffStepNum,'Value',1);
%       set(handles.plotBleachStepOption_greaterThan,'Value',0);
        if size(handles.old_1st_Results,2)>2 %not previous version's analysis data
        oldBin=[num2str(handles.old_1st_Results(currentRoi,3)),'-',...
                num2str(handles.old_2nd_Results(currentRoi,3)),'-',...
                num2str(handles.old_3rd_Results(currentRoi,3))];
        set(handles.tb_oldBinSize,'String',oldBin);
   end


% --- Executes on button press in plotBleachStepOption_greaterThan.
```

```
function plotBleachStepOption_greaterThan_Callback(hObject, eventdata, handles)
% hObject    handle to plotBleachStepOption_greaterThan (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of plotBleachStepOption_greaterThan
if get(hObject,'Value')
   set(handles.plotOnlyBleachStepOption,'Value',0);
   set(handles.cb_plotGraphWithDiffStepNum,'Value',0);
else
   set(handles.plotOnlyBleachStepOption,'Value',1);
end
guidata(hObject, handles);


% --- Executes on button press in cb_plotGraphWithDiffStepNum.
function cb_plotGraphWithDiffStepNum_Callback(hObject, eventdata, handles)
% hObject    handle to cb_plotGraphWithDiffStepNum (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of cb_plotGraphWithDiffStepNum

if get(hObject,'Value')
   set(handles.plotOnlyBleachStepOption,'Value',0);
   set(handles.plotBleachStepOption_greaterThan,'Value',0);
else
   set(handles.plotOnlyBleachStepOption,'Value',1);
end
guidata(hObject, handles);


% --- Executes on button press in NextGraphPlotAddTwoFrames.
function NextGraphPlotAddTwoFrames_Callback(hObject, eventdata, handles)
% hObject    handle to NextGraphPlotAddTwoFrames (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of NextGraphPlotAddTwoFrames


% --- Executes on button press in plotOnlyBleachStepOption.
function plotOnlyBleachStepOption_Callback(hObject, eventdata, handles)
% hObject    handle to plotOnlyBleachStepOption (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hint: get(hObject,'Value') returns toggle state of plotOnlyBleachStepOption
if get(hObject,'Value')
   set(handles.plotBleachStepOption_greaterThan,'Value',0);
   set(handles.cb_plotGraphWithDiffStepNum,'Value',0);
else
   set(handles.plotBleachStepOption_greaterThan,'Value',1);
end
guidata(hObject, handles);


function plotOnlyBleachStepInput_Callback(hObject, eventdata, handles)
% hObject    handle to plotOnlyBleachStepInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of plotOnlyBleachStepInput as text
%        str2double(get(hObject,'String')) returns contents of plotOnlyBleachStepInput as a
double


% --- Executes during object creation, after setting all properties.
function plotOnlyBleachStepInput_CreateFcn(hObject, eventdata, handles)
% hObject    handle to plotOnlyBleachStepInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if                  ispc                &&                  isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in verticalLine.
function verticalLine_Callback(hObject, eventdata, handles)
% hObject    handle to verticalLine (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of verticalLine

function verticalLineInput_Callback(hObject, eventdata, handles)
% hObject    handle to verticalLineInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of verticalLineInput as text
%        str2double(get(hObject,'String')) returns contents of verticalLineInput as a double


% --- Executes during object creation, after setting all properties.
function verticalLineInput_CreateFcn(hObject, eventdata, handles)
% hObject    handle to verticalLineInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in filterTimeTrace.
function filterTimeTrace_Callback(hObject, eventdata, handles)
% hObject    handle to filterTimeTrace (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of filterTimeTrace
   if ~get(hObject,'Value')
      data=handles.plottedData;
      plotSimpleGraph2(handles, data);
   else
      plotSimpleGraph2(handles);
   end

   guidata(hObject, handles);

   function handles= LoadADataFile(handles,fileName,path,inputRoiNum)
      if get(handles.hideFileName,'Value')==0
         set(handles.fileName,'String',fileName);

      else
         set(handles.fileName,'String','');
      end
      cd(path); %make the folder my current directory so that it will dircet you here later
      hWaitbar = waitbar(0,'Populating raw data');
      %update fileNum1 to use in saving data
      %get data from the file
```

```
        fullFileName=fullfile(path,fileName); % Make a new file name with path name in
front of the file name.
        [data,~]=importdata(fullFileName); %get the file

        %remove frameNum, mean and average data column from raw data table
        if(strcmp(data.colheaders{1, 2},'Average'))==1 %strcmp does not give any error if
two string are of different size
           rawData=data.data(:,4:end);
        elseif (strcmp(data.colheaders{1, end-1},'Average'))==1
           rawData=data.data(:,2:end-2);
        else
           rawData=data.data(:,2:end);
        end
%        handles.actualRawData=rawData;
        numFrameBeforeCameraSutterOpened=0;
        for i=2:size(rawData,1)
           if rawData(i,1)==rawData(1,1)

numFrameBeforeCameraSutterOpened=numFrameBeforeCameraSutterOpened+5;
           else
              break;
           end
        end
rawData=rawData(numFrameBeforeCameraSutterOpened+1:end,:);
handles.datafilePathname=path; %same as filename

        handles.rawData=rawData;

       %following statement for "previous graph"
        if nargin==3
           currentRoi=1;
            handles.outputData=500*ones(size(rawData,2),4);%to get ROInum and stepNum
        else
          currentRoi=inputRoiNum;
           handles.outputData=handles.lastDataOutputData;
        end
        set(handles.roiNum,'String',num2str(currentRoi));

        handles=plotSimpleGraph_newApproach(handles) ;
        %display on

        handles.old_1st_Results=[]; %sothat data from previous analysis does not show
        handles.old_2nd_Results=[]; %sothat data from previous analysis does not show
        handles.old_3rd_Results=[];
        %display total graph num
    set(handles.tb_totalGraph,'String',num2str(size(rawData,2)));
```

```
        close(hWaitbar);

       function
resultFileNameCorspndToDataFileName=getCorFileName(dataFileName,allResultFileN
ames)

          dataFileName=dataFileName(1:end-5); %remove .xlsx

          for i=1:length(allResultFileNames)
             aResultFileName=allResultFileNames{i};
             aResultFileName=aResultFileName(11:end-4); %remove "Result of- " from file
name and ".txt"
             if strcmp(dataFileName,aResultFileName)
                resultFileNameCorspndToDataFileName=allResultFileNames{i};
                break;

             elseif i==length(allResultFileNames)
                resultFileNameCorspndToDataFileName={};
             end
          end

% --- Executes on button press in fit_timeTrace.
function fit_timeTrace_Callback(hObject, eventdata, handles)
% hObject    handle to fit_timeTrace (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of fit_timeTrace
plotSimpleGraph2(handles,handles.plottedData);

 guidata(hObject, handles);

function    plotOnlyBleachStepOption_greaterThanInput_Callback(hObject,    eventdata,
handles)
% hObject    handle to plotOnlyBleachStepOption_greaterThanInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%          Hints:          get(hObject,'String')          returns          contents          of
plotOnlyBleachStepOption_greaterThanInput as text
%                              str2double(get(hObject,'String'))    returns    contents    of
plotOnlyBleachStepOption_greaterThanInput as a double


% --- Executes during object creation, after setting all properties.
function plotOnlyBleachStepOption_greaterThanInput_CreateFcn(hObject, ~, handles)
```

```
% hObject    handle to plotOnlyBleachStepOption_greaterThanInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in displayOldResult.
function displayOldResult_Callback(hObject, eventdata, handles)
% hObject    handle to displayOldResult (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of displayOldResult
if get(hObject,'Value')
    set(handles.uipanel_DisplayOldResults,'Visible','on')
    set(get(handles.uipanel_DisplayOldResults,'Children'), 'Visible','on')
      if isfield (handles,'old_3rd_Results') && ~isempty(handles.old_3rd_Results);
         handles=plot_3rdResult(handles); %will plot 2nd and 1st as well
      elseif isfield (handles,'old_2nd_Results') && ~isempty(handles.old_2nd_Results);
         handles=plot_2ndResult(handles); %will pot 1st as well
      elseif isfield (handles,'old_1st_Results') && ~isempty(handles.old_1st_Results);
         handles=plot_1stResult(handles);
      end
else
    set(handles.uipanel_DisplayOldResults,'Visible','off')
    set(get(handles.uipanel_DisplayOldResults,'Children'), 'Visible','off')
end

guidata(hObject, handles);

function calculate1_Callback(hObject, eventdata, handles)
% hObject    handle to calculate1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of calculate1 as text
%        str2double(get(hObject,'String')) returns contents of calculate1 as a double
input=get(hObject,'String');
[x1, x2]=getValues(input);
%get the data
```

```
data=handles.plottedData;
tempData=data(x1:x2,1);
mean1=round(mean(tempData));
std1=round(std(tempData));

set(handles.mean1,'String',num2str(mean1));
set(handles.std1,'String',num2str(std1));
calculate_1Data=[(x1:x2)',mean1*ones(size(tempData))];
%for calculating mean and std, romove ealir data


%plot options
%only this has data
if ~isempty(handles.calculate_2Data)==0 && ~isempty(handles.calculate_3Data)==0
   plotIt(handles,data,calculate_1Data);

elseif ~isempty(handles.calculate_2Data)==1 && ~isempty(handles.calculate_3Data)==0
   plotIt(handles,data,calculate_1Data,handles.calculate_2Data);
   mean2=str2double(get(handles.mean2,'String'));
   set(handles.meanDiff1,'String',num2str(abs(mean1-mean2)));
elseif ~isempty(handles.calculate_2Data)==0 && ~isempty(handles.calculate_3Data)==1
   plotIt(handles,data,calculate_1Data,handles.calculate_3Data);
   mean3=str2double(get(handles.mean3,'String'));
   set(handles.meanDiff3,'String',num2str(abs(mean1-mean3)));
else
   plotIt(handles,data,calculate_1Data,handles.calculate_2Data,handles.calculate_3Data);
   mean2=str2double(get(handles.mean2,'String'));
   set(handles.meanDiff1,'String',num2str(abs(mean1-mean2)));
   mean3=str2double(get(handles.mean3,'String'));
   set(handles.meanDiff3,'String',num2str(abs(mean1-mean3)));
end

handles.calculate_1Data=calculate_1Data;
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function calculate1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to calculate1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if                ispc                &&                isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
   set(hObject,'BackgroundColor','white');
end



function calculate2_Callback(hObject, eventdata, handles)
% hObject    handle to calculate2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of calculate2 as text
%        str2double(get(hObject,'String')) returns contents of calculate2 as a double
input=get(hObject,'String');
[x1, x2]=getValues(input);
%get the data
data=handles.plottedData;
tempData=data(x1:x2,1);
mean2=round(mean(tempData));
std2=round(std(tempData));

set(handles.mean2,'String',num2str(mean2));
set(handles.std2,'String',num2str(std2));
calculate_2Data=[(x1:x2)',mean2*ones(size(tempData))];

%plot option

if ~isempty(handles.calculate_1Data)==0 && ~isempty(handles.calculate_3Data)==0
   plotIt(handles,data,calculate_2Data);

elseif ~isempty(handles.calculate_1Data)==1 && ~isempty(handles.calculate_3Data)==0
   plotIt(handles,data,handles.calculate_1Data,calculate_2Data);
   mean1=str2double(get(handles.mean1,'String'));
   set(handles.meanDiff1,'String',num2str(abs(mean2-mean1)));
elseif ~isempty(handles.calculate_1Data)==0 && ~isempty(handles.calculate_3Data)==1
   plotIt(handles,data,calculate_2Data,handles.calculate_3Data);
   mean3=str2double(get(handles.mean3,'String'));
   set(handles.meanDiff2,'String',num2str(abs(mean2-mean3)));
else
   plotIt(handles,data,handles.calculate_1Data,calculate_2Data,handles.calculate_3Data);
   mean1=str2double(get(handles.mean1,'String'));
   set(handles.meanDiff1,'String',num2str(abs(mean2-mean1)));
   mean3=str2double(get(handles.mean3,'String'));
   set(handles.meanDiff2,'String',num2str(abs(mean2-mean3)));
end

%set(handles.meanDiff1,'String',num2str(abs(mean1-handles.mean2)));
```

```
handles.calculate_2Data=calculate_2Data;
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function calculate2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to calculate2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

function calculate3_Callback(hObject, eventdata, handles)
% hObject    handle to calculate3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of calculate3 as text
%        str2double(get(hObject,'String')) returns contents of calculate3 as a double
input=get(hObject,'String');
[x1, x2]=getValues(input);
%get the data
data=handles.plottedData;
tempData=data(x1:x2,1);
mean3=round(mean(tempData));
std3=round(std(tempData));

set(handles.mean3,'String',num2str(mean3));
set(handles.std3,'String',num2str(std3));
calculate_3Data=[(x1:x2)',mean3*ones(size(tempData))];

%plot option
if ~isempty(handles.calculate_1Data)==0 && ~isempty(handles.calculate_2Data)==0
   plotIt(handles,data,calculate_3Data);

elseif ~isempty(handles.calculate_1Data)==1 && ~isempty(handles.calculate_2Data)==0
   plotIt(handles,data,handles.calculate_1Data,calculate_3Data);
   mean1=str2double(get(handles.mean1,'String'));
   set(handles.meanDiff3,'String',num2str(abs(mean3-mean1)));
elseif ~isempty(handles.calculate_1Data)==0 && ~isempty(handles.calculate_2Data)==1
   plotIt(handles,data,handles.calculate_2Data,calculate_3Data);
```

```
    mean2=str2double(get(handles.mean2,'String'));
    set(handles.meanDiff2,'String',num2str(abs(mean3-mean2)));
else
    plotIt(handles,data,handles.calculate_1Data,handles.calculate_2Data,calculate_3Data);
    mean1=str2double(get(handles.mean1,'String'));
    set(handles.meanDiff3,'String',num2str(abs(mean3-mean1)));
    mean2=str2double(get(handles.mean2,'String'));
    set(handles.meanDiff2,'String',num2str(abs(mean3-mean2)));
end

handles.calculate_3Data=calculate_3Data;
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function calculate3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to calculate3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


  function [x1, x2]=getValues(input)

%seperate the numbers from input
if (strcmp(input(1,2),'-'))==1;
    x1=str2double(input(1,1));
    x2=str2double(input(1,3:end));
elseif (strcmp(input(1,3),'-'))==1;
    x1=str2double(input(1,1:2));
    x2=str2double(input(1,4:end));
elseif (strcmp(input(1,4),'-'))==1;
    x1=str2double(input(1,1:3));
    x2=str2double(input(1,5:end));
else
    uiwait(msgbox('Remember you need to put two numbers separated by a
hipen','Error','error'));
    return;
end

%check if x1 or x2 bigger
```

```
if x1>x2
   a1=x2;
   a2=x1;
   x1=a1;
   x2=a2;
end

% --- Executes on selection change in NumPontsAddToPlot.
function NumPontsAddToPlot_Callback(hObject, eventdata, handles)
% hObject    handle to NumPontsAddToPlot (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns NumPontsAddToPlot contents as
cell array
%        contents{get(hObject,'Value')} returns selected item from NumPontsAddToPlot
% set(handles.NumPontsAddToPlot,'Value',2)
handles=plotSimpleGraph_newApproach(handles) ;
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function NumPontsAddToPlot_CreateFcn(hObject, eventdata, handles)
% hObject    handle to NumPontsAddToPlot (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if                  ispc                &&                  isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in displayVeticalLinesforMeanAndSTD.
function displayVeticalLinesforMeanAndSTD_Callback(hObject, eventdata, handles)
% hObject    handle to displayVeticalLinesforMeanAndSTD (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles=plotSimpleGraph_newApproach(handles); % necessary to avoid multiple plots of
lines on the same graphs

handles=plotLines(handles);
guidata(hObject,handles);
```

```
   function handles=plotLines(handles,line_1,line_2,line_3)
  % Hint: get(hObject,'Value') returns toggle state of displayVeticalLinesforMeanAndSTD
 if nargin<2
  handles.edge=5;
 handles.secondFromRihgtLineDist=10;
line_1=handles.edge;
xLimit=xlim(handles.axes1);
line_2=floor(xLimit(2)/4*3);
line_3=floor(xLimit(2)-handles.secondFromRihgtLineDist);
% line_4=floor(xLimit(2)-handles.edge);
 end
yLimit=ylim(handles.axes1);

% --- Executes on button press in calculate.
function calculate_Callback(hObject, eventdata, handles)
% hObject    handle to calculate (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% replotLines(hObject, eventdata, handles)
handles=plotSimpleGraph_newApproach(handles) ;
guidata(hObject,handles);

function handles=plotSimpleGraph_newApproach(handles)

      currentRoi=str2double(get(handles.roiNum,'String')); %get the graph number which
is actually ROI number
      data=handles.rawData(:,currentRoi);
      %conts = get(handles.NumPontsAddToPlot,'String');
      NumPontsAddToPlot_input = get(handles.NumPontsAddToPlot,'Value');
      combinedData=zeros(floor(size(data,1)/NumPontsAddToPlot_input),1);
      for r =1:size(combinedData,1)
        combinedData(r,1)=mean(data(NumPontsAddToPlot_input*(r-
1)+1:r*NumPontsAddToPlot_input));
      end
      %to plot a base line, plot mean of last 20 frames
      if size(combinedData,1)>25
         baseData=mean(combinedData(end-20:end,1))*ones(size(combinedData,1),1);
      else
         uiwait(msgbox({'"How Many Frames to plot?" need to be greater than 30'}),1);
         return
      end
%       cla reset;

         if get(handles.verticalLine,'Value')
           xVal=str2double(get(handles.verticalLineInput,'String'));
           xVal=round(xVal/NumPontsAddToPlot_input);
```

```
            else
               xVal=0;
            end
        temp=round(min(combinedData(:,1))):round(max(combinedData(:,1)));
        %since when we plot any graph handles assocated with lines
        %(h_k_line, h_g_line, etc) disappeared, we are collecting those
        %numbers here
  if get(handles.calculate,'Value')==0
            axes(handles.axes1)
        cla reset;
        xAxisLimit=str2double(get(handles.plotOpt_frameNum,'String'));
        plot(1:size(combinedData(:,1),1),combinedData(:,1),'b','LineWidth',1);
        hold on
       plot(1:size(combinedData(:,1),1),combinedData(:,1),'b',...
           1:size(baseData(:,1),1),baseData(:,1),'--r',...
           xVal*ones(length(temp),1),temp,'--k','LineWidth',1);
        hold off
         %display x limit values
        if size(combinedData,1)<xAxisLimit
            xAxisLimit=size(combinedData,1);
        end
        xlim([0 xAxisLimit]);
    %      handles.plottedData=data;
         xlabel('Number of Frames'); ylabel('Average Intensity');
         %display vertical lines for finding mean and std
        %when calculate button was clicked
  else
  %get line's x coordinates limits
        line_1 =round(get(handles.h_k_line, 'XData'));
        line_2=round(get(handles.h_r_line, 'XData'));
        line_3=round(get(handles.h_g_line, 'XData'));
%         line_4=round(get(handles.h_m_line, 'XData'));
        %get one values form the cooridate limits
        line_1=line_1(1);
        line_2=line_2(1);
        line_3=line_3(1);
        xLimit=xlim(handles.axes1);
        xAxisLimit =xLimit(2);

        if line_1<1 ||  line_2<1 ||  line_3<1
            if line_1<1
               line_1=1;
            elseif line_2<1
               line_2=1;
            else
               line_3=1;
```

```
        end


    elseif line_1>xAxisLimit ||  line_2>xAxisLimit ||  line_3>xAxisLimit
        if line_1>xAxisLimit
          line_1=xAxisLimit;
        elseif line_2>xAxisLimit
          line_2=xAxisLimit;
        else
          line_3=xAxisLimit;
        end
    end
%       line_4=line_4(1);
    %sort the x coordinates
    xValues=sort([line_1,line_2,line_3]);
    line_1=xValues(1);
    line_2=xValues(2);
    line_3=xValues(3);
%       line_4=xValues(4);
    xLimitVal=xlim(handles.axes1);
    xLimitVal=xLimitVal(2);
    plotedData=combinedData(1:xLimitVal,1);
 %plot graphs agains
    axes(handles.axes1)
    cla reset;
    xAxisLimit=str2double(get(handles.plotOpt_frameNum,'String'));
    plot(1:size(combinedData(:,1),1),combinedData(:,1),'b','LineWidth',1);
    hold on
   plot(1:size(combinedData(:,1),1),combinedData(:,1),'b',...
      1:size(baseData(:,1),1),baseData(:,1),'--r',...
      xVal*ones(length(temp),1),temp,'--k','LineWidth',1);
    hold off
     %display x limit values
    if size(combinedData,1)<xAxisLimit
       xAxisLimit=size(combinedData,1);
    end
    xlim([0 xAxisLimit]);
  %     handles.plottedData=data;
     xlabel('Number of Frames'); ylabel('Average Intensity');

     handles=plotLines(handles,line_1,line_2,line_3);

%get mean Intensity and Bleaching time of first interval
    data_12=plotedData(line_1:line_2);
    data_23=plotedData(line_2:line_3);
```

```
        mean_12=round(mean(data_12));
        mean_23=round(mean(data_23));

        std_12=round(std(data_12));
        std_23=round(std(data_23));

        hold on
        axes(handles.axes1)
        plot(line_1:line_2,mean_12*(ones(1, size(data_12,1))),'m',...
           line_2:line_3,mean_23*(ones(1, size(data_23,1))), 'm', 'LineWidth', 4)
        hold off

        %set values
        set(handles.avg_1,'String',num2str(mean_12));
        set(handles.avg_2,'String',num2str(mean_23));

        set(handles.avgDif,'String',num2str(round((abs(mean_12-mean_23)))));

        set(handles.std_1,'String',num2str(std_12));
        set(handles.std_2,'String',num2str(std_23));

        set(handles.ratio_1,'String',num2str(round(abs(mean_12-mean_23)/std_12,2)));
        set(handles.ratio_2,'String',num2str(round(abs(mean_12-mean_23)/std_23,2)));
  end
  handles.plottedData =combinedData;

  function
nextRoi=getNextRoi_greaterThan(stepNum,currentRoi,old_1st_Results,old_2nd_Results
,old_3rd_Results)
if nargin<4
    for i=currentRoi+1:size(old_1st_Results,1)
        if old_1st_Results(i,2)>=stepNum
          break;
        end
    end
elseif nargin<5
    for i=currentRoi+1:size(old_1st_Results,1)
        if  old_1st_Results(i,2)>=stepNum || old_2nd_Results(i,2)>=stepNum; %al least one
is greater than the value
          break;
        end
    end
else
    for i=currentRoi+1:size(old_1st_Results,1)
        if        old_1st_Results(i,2)>=stepNum        ||old_2nd_Results(i,2)>=stepNum        ||
old_3rd_Results(i,2)>=stepNum
```

```
        break;
      end
   end

end
nextRoi=i;


function
perviousRoi=getPreviousRoi_greaterThan(stepNum,currentRoi,old_1st_Results,old_2nd
_Results,old_3rd_Results)
if nargin<4
   for i=currentRoi-1:-1:1
      if old_1st_Results(i,2)>=stepNum
        break;
      end
   end
elseif nargin<5
   for i=currentRoi-1:-1:1
      if old_1st_Results(i,2)>=stepNum || old_2nd_Results(i,2)>=stepNum; %al least one
is greater than the value
        break;
      end
   end
else
   for i=currentRoi-1:-1:1
      if      old_1st_Results(i,2)>=stepNum      ||old_2nd_Results(i,2)>=stepNum      ||
old_3rd_Results(i,2)>=stepNum
        break;
      end
   end
end
perviousRoi=i;


function                                                    nextRoi=getNextRoi_aValue(aVal,
currentRoi,old_1st_Results,old_2nd_Results,old_3rd_Results)
if nargin<4

   for i=currentRoi+1:size(old_1st_Results,1)
      if old_1st_Results(i,2)==aVal
        break;
      end
   end
elseif nargin<5
   for i=currentRoi+1:size(old_1st_Results,1)
```

```
      if  old_1st_Results(i,2)==aVal || old_2nd_Results(i,2)==aVal;
      break;
      end
   end
else
   for i=currentRoi+1:size(old_1st_Results,1)
      if old_1st_Results(i,2)==aVal || old_2nd_Results(i,2)==aVal ||
old_3rd_Results(i,2)==aVal;
        break;
      end
   end
end
nextRoi=i;


 function nextRoi=getPreviousRoi_aValue(aVal,
currentRoi,old_1st_Results,old_2nd_Results,old_3rd_Results)


if nargin<4

   for i=currentRoi-1:-1:1
      if old_1st_Results(i,2)==aVal
        break;
      end
   end

elseif nargin<5
   for i=currentRoi-1:-1:1
%         nextRoi=i;
%                         if ~isequal(old_1st_Results(i,2),old_2nd_Results(i,2)) &&
any([old_1st_Results(i,2),old_2nd_Results(i,2)]); %if not equal and has atleast one
nonezero value
      if  old_1st_Results(i,2)==aVal || old_2nd_Results(i,2)==aVal;
      break;
      end
   end

else
   for i=currentRoi-1:-1:1
      if old_1st_Results(i,2)==aVal || old_2nd_Results(i,2)==aVal ||
old_3rd_Results(i,2)==aVal;
%         nextRoi=i;
        break;
      end
   end
```

```
end

nextRoi=i;

function
nextRoi=getNextRoi_differntStepNum(currentRoi,old_1st_Results,old_2nd_Results,old_
3rd_Results)


if nargin<3

   for i=currentRoi+1:size(old_1st_Results,1)
      if old_1st_Results(i,2)>0
        break;
      end
   end



elseif nargin<4
   for i=currentRoi+1:size(old_1st_Results,1)
%         nextRoi=i;
      if                    ~isequal(old_1st_Results(i,2),old_2nd_Results(i,2))          &&
any([old_1st_Results(i,2),old_2nd_Results(i,2)]); %if not equal and has atleast one
nonezero value
        break;
      end
   end

else
   for i=currentRoi+1:size(old_1st_Results,1)
      if    ~isequal(old_1st_Results(i,2),old_2nd_Results(i,2),old_3rd_Results(i,2))   &&
any([old_1st_Results(i,2),old_2nd_Results(i,2),old_3rd_Results(i,2)]);
%         nextRoi=i;
        break;
      end
   end

end

nextRoi=i;

function
perviousRoi=getPreviousRoi_differntStepNum(currentRoi,old_1st_Results,old_2nd_Res
ults,old_3rd_Results)
```

```
if nargin<3

   for i=currentRoi-1:-1:1
      if old_1st_Results(i,2)>0
        break;
      end
   end



elseif nargin<4
   for i=currentRoi-1:-1:1
      if ~isequal(old_1st_Results(i,2),old_2nd_Results(i,2)) &&
any([old_1st_Results(i,2),old_2nd_Results(i,2)]); %if not equal and has atleast one
nonezero value
        break;
      end
   end

else
   for i=currentRoi-1:-1:1
      if ~isequal(old_1st_Results(i,2),old_2nd_Results(i,2),old_3rd_Results(i,2)) &&
any([old_1st_Results(i,2),old_2nd_Results(i,2),old_3rd_Results(i,2)]);
         break;
      end
   end

end

perviousRoi=i;

% --- Executes on button press in noSteps.
function noSteps_Callback(hObject, eventdata, handles)
% hObject    handle to noSteps (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','');
guidata(hObject, handles);

% --- Executes on button press in zero.
function zero_Callback(hObject, eventdata, handles)
% hObject    handle to zero (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
set(handles.stepNum,'String','0');
guidata(hObject, handles);


% --- Executes on button press in one.
function one_Callback(hObject, eventdata, handles)
% hObject    handle to one (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','1');
guidata(hObject, handles);


% --- Executes on button press in two.
function two_Callback(hObject, eventdata, handles)
% hObject    handle to two (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','2');
guidata(hObject, handles);

% --- Executes on button press in three.
function three_Callback(hObject, eventdata, handles)
% hObject    handle to three (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','3');
guidata(hObject, handles);

% --- Executes on button press in five.
function five_Callback(hObject, eventdata, handles)
% hObject    handle to five (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','5');
guidata(hObject, handles);

% --- Executes on button press in six.
function six_Callback(hObject, eventdata, handles)
% hObject    handle to six (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','6');
guidata(hObject, handles);

% --- Executes on button press in four.
```

```
function four_Callback(hObject, eventdata, handles)
% hObject    handle to four (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','4');
guidata(hObject, handles);


% --- Executes on button press in seven.
function seven_Callback(hObject, eventdata, handles)
% hObject    handle to seven (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','7');
guidata(hObject, handles);

function eight_Callback(hObject, eventdata, handles)
% hObject    handle to seven (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','8');
guidata(hObject, handles);

function nine_Callback(hObject, eventdata, handles)
% hObject    handle to seven (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.stepNum,'String','9');
guidata(hObject, handles);


% --- Executes on button press in hideFileName.
function hideFileName_Callback(hObject, eventdata, handles)
% hObject    handle to hideFileName (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
 if get(handles.hideFileName,'Value')==1
         set(handles.fileName,'String','');

 else
    set(handles.fileName,'String',handles.fileNames{handles.curFileNum});
 end

 guidata(hObject, handles);
% Hint: get(hObject,'Value') returns toggle state of hideFileName
```

```
    %% this block of code was modified and directy adpoted from following copyright
sources
    %https://www.mathworks.com/matlabcentral/fileexchange/42284-drag-line-in-gui
% % Copyright (c) 2013, Gero Nootz
% % All rights reserved.
% %
% % Redistribution and use in source and binary forms, with or without
% % modification, are permitted provided that the following conditions are
% % met:
% %
% %     * Redistributions of source code must retain the above copyright
% %       notice, this list of conditions and the following disclaimer.
% %     * Redistributions in binary form must reproduce the above copyright
% %       notice, this list of conditions and the following disclaimer in
% %       the documentation and/or other materials provided with the distribution
% % THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND
CONTRIBUTORS "AS IS"
% % AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE
% % IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE
% % ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE
% % LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR
% % CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF
% % SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
BUSINESS
% % INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN
% % CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE)
% % ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
ADVISED OF THE
% % POSSIBILITY OF SUCH DAMAGE.

   % Plot lines
   handles.h_k_line = line([line_1 line_1], [yLimit(1) yLimit(2)],...
      'LineWidth', 2, 'Color', 'k', 'ButtonDownFcn',
@(hObject,eventdata)simpleGraph_byFaruk('startDragK_Fcn',hObject,eventdata,guidata(
hObject)));

   handles.h_r_line = line([line_2 line_2], [yLimit(1) yLimit(2)],...
```

```
    'LineWidth', 2, 'Color', 'r', 'ButtonDownFcn',
@(hObject,eventdata)simpleGraph_byFaruk('startDragR_Fcn',hObject,eventdata,guidata(
hObject)));

   handles.h_g_line = line([line_3 line_3], [yLimit(1) yLimit(2)],...
    'LineWidth', 2, 'Color', 'g', 'ButtonDownFcn',
@(hObject,eventdata)simpleGraph_byFaruk('startDragG_Fcn',hObject,eventdata,guidata(
hObject)));

% %    handles.h_m_line = line([line_4 line_4], [yLimit(1) yLimit(2)],...
% %                'LineWidth', 2, 'Color', 'm', 'ButtonDownFcn',
@(hObject,eventdata)simpleGraph_byFaruk('startDragM_Fcn',hObject,eventdata,guidata
(hObject)));
% % %%



function startDragK_Fcn(hObject, eventdata, handles)
    set(handles.figure1, 'WindowButtonMotionFcn',
@(hObject,eventdata)simpleGraph_byFaruk('draggingK_Fcn',hObject,eventdata,guidata(
hObject)));


function draggingK_Fcn(hObject, eventdata, handles)
    pt = get(handles.axes1, 'CurrentPoint');
    set(handles.h_k_line, 'XData', pt(1)*[1 1]);



function startDragR_Fcn(hObject, eventdata, handles)
    set(handles.figure1, 'WindowButtonMotionFcn',
@(hObject,eventdata)simpleGraph_byFaruk('draggingR_Fcn',hObject,eventdata,guidata(
hObject)));

function draggingR_Fcn(hObject, eventdata, handles)
    pt = get(handles.axes1, 'CurrentPoint');
    set(handles.h_r_line, 'XData', pt(1)*[1 1]);


function startDragG_Fcn(hObject, eventdata, handles)
    set(handles.figure1, 'WindowButtonMotionFcn',
@(hObject,eventdata)simpleGraph_byFaruk('draggingG_Fcn',hObject,eventdata,guidata(
hObject)));

function draggingG_Fcn(hObject, eventdata, handles)
```

```
    pt = get(handles.axes1, 'CurrentPoint');
    set(handles.h_g_line, 'XData', pt(1)*[1 1]);


    function startDragM_Fcn(hObject, eventdata, handles)
    set(handles.figure1, 'WindowButtonMotionFcn',
@(hObject,eventdata)simpleGraph_byFaruk('draggingM_Fcn',hObject,eventdata,guidata(
hObject)));

function draggingM_Fcn(hObject, eventdata, handles)
    pt = get(handles.axes1, 'CurrentPoint');
    set(handles.h_m_line, 'XData', pt(1)*[1 1]);



function stopDragFcn(hObject, eventdata, handles)
    set(handles.figure1, 'WindowButtonMotionFcn', '');
```

Outline of the GUI: StiochiometryFitting_byFaruk.fig

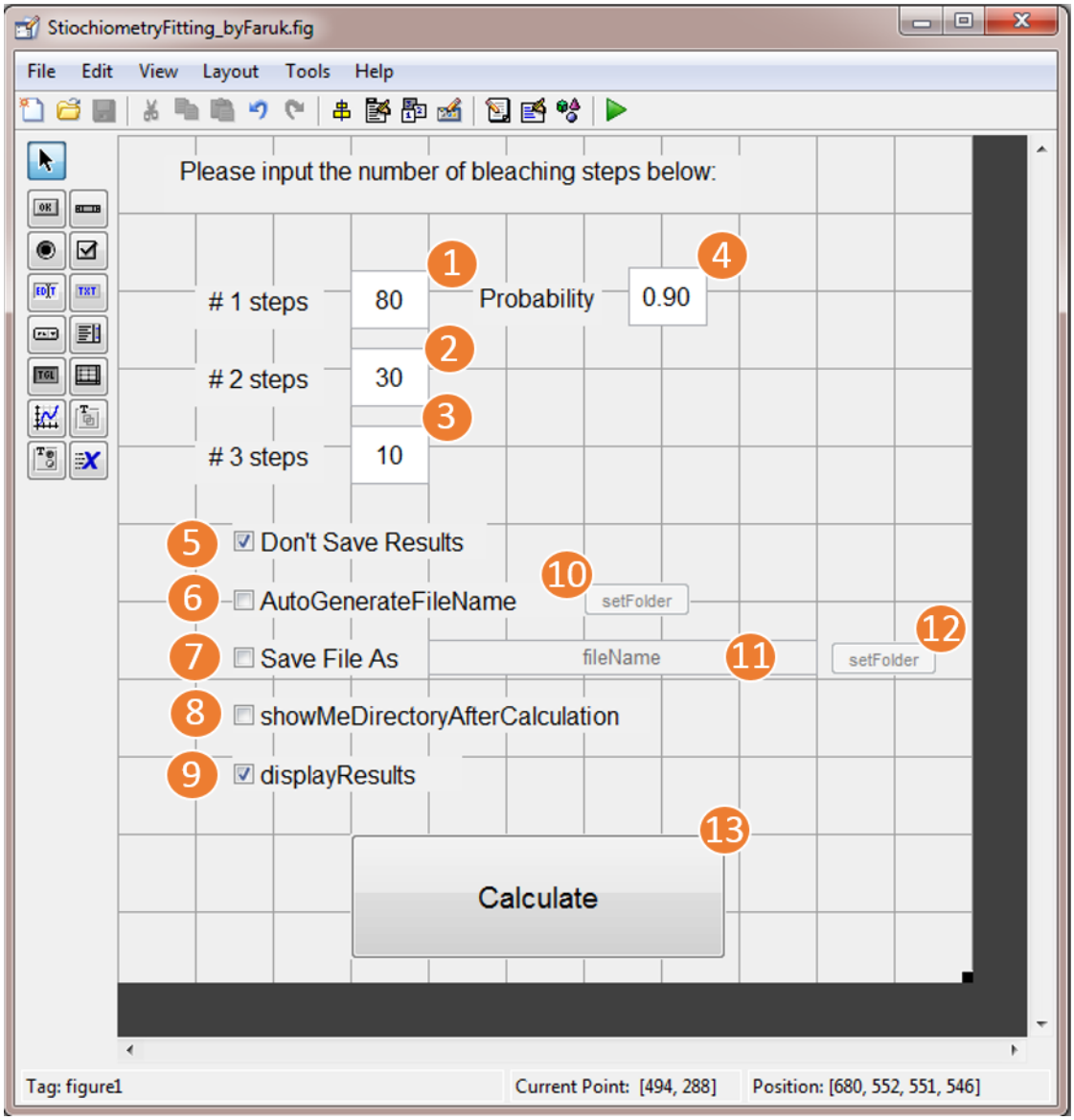

Figure S5: Outline of the GUI used to fit photobleaching distribution to obtain proportion of two and three labeled subunits. This GUI was saved as "StiochiometryFitting_byFaruk.fig". The underlying variables (Tag) for components 1-13 can be found in the Table S3.

Table S3: The essential variables used in the figure for GUI (Figure S5). All other components were either Static Texts or Panels.

Number in Figure S1	Type of component	String	Tag
1	Edit Text	80	num_1steps
2	Edit Text	30	num_2steps
3	Edit Text	10	num_3steps
4	Edit Text	0.90	probability
5	Check Box	Don't Save Results	donotSaveResults
6	Check Box	AutoGenerateFileName	AutoGenerateFileName
7	Check Box	Save File As	saveAs
8	Check Box	showMeDirectoryAfterCalculation	showMeDirectoryAfterCalculation
9	Check Box	displayResults	displayResults
10	Push Button	setFolder	setFolder1
11	Edit Text	fileNameInput	fileNameInput
12	Push Button	setFolder	setFolder2
13	Push Button	Calculate	calculate

Matlab Code in a file called, “StiochiometryFitting_byFaruk.m”

```
function varargout = StiochiometryFitting_byFaruk(varargin)

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @StiochiometryFitting_byFaruk_OpeningFcn, ...
                   'gui_OutputFcn',  @StiochiometryFitting_byFaruk_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before StiochiometryFitting_byFaruk is made visible.
function StiochiometryFitting_byFaruk_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to StiochiometryFitting_byFaruk (see VARARGIN)

% Choose default command line output for StiochiometryFitting_byFaruk
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes StiochiometryFitting_byFaruk wait for user response (see
UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
```

```
function varargout = StiochiometryFitting_byFaruk_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


function num_1steps_Callback(hObject, eventdata, handles)
% hObject    handle to num_1steps (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of num_1steps as text
%        str2double(get(hObject,'String')) returns contents of num_1steps as a double
% contents = cellstr(get(handles.popupmenu1,'String')) ;
% cond=contents{get(handles.popupmenu1,'Value')} ;
% if strcmp(cond, 'Fit 1 and 2 labeled xFP')
%     num_1steps=str2double(get(hObject,'String'));
%     set(handles.num_2steps,'String',num2str(100-num_1steps));
% end

% guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function num_1steps_CreateFcn(hObject, eventdata, handles)
% hObject    handle to num_1steps (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end


function num_2steps_Callback(hObject, eventdata, handles)
% hObject    handle to num_2steps (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of num_2steps as text
%        str2double(get(hObject,'String')) returns contents of num_2steps as a double
% --- Executes during object creation, after setting all properties.
function num_2steps_CreateFcn(hObject, eventdata, handles)
% hObject    handle to num_2steps (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

function num_3steps_Callback(hObject, eventdata, handles)
% hObject    handle to num_3steps (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of num_3steps as text
%        str2double(get(hObject,'String')) returns contents of num_3steps as a double

% --- Executes during object creation, after setting all properties.
function num_3steps_CreateFcn(hObject, eventdata, handles)
% hObject    handle to num_3steps (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in calculate.
function calculate_Callback(hObject, eventdata, handles)
% hObject    handle to calculate (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
hb=waitbar(0,'Please wait...');

fit_2_3_stoiochiometry (handles)
```

```
close(hb);
guidata(hObject, handles);

function fit_2_3_stoiochiometry (handles)

num_1steps = str2double(get(handles.num_1steps,'String'));
num_2steps = str2double(get(handles.num_2steps,'String'));
num_3steps = str2double(get(handles.num_3steps,'String'));
prob = str2double(get(handles.probability,'String'));

%get the ratio of the number of steps
obsDist=[num_1steps, num_2steps, num_3steps]; %an arry of the number of steps
totalSteps=sum(obsDist);
obsDistRatio=obsDist/totalSteps; %observed distribution array
errorInObsDistRatio=sqrt(obsDist)/totalSteps; %an array of error in the observed distribution.

%get bionomal distributions
x=[1 2 3];
distFrom_2LabeledSub=binopdf(x,2,prob);
distFrom_3LabeledSub=binopdf(x,3,prob);
%normailize as there is a possiblity to observe 0 steps
normDistFrom_2LabeledSub=distFrom_2LabeledSub/sum(distFrom_2LabeledSub);
normDistFrom_3LabeledSub=distFrom_3LabeledSub/sum(distFrom_3LabeledSub);
%make a matrix to weight the binomal distribution and get a final
%distribution
a1 =(0:100)';
a2=100-a1;
weightMat=[a1,a2]/100; % the weight matrix with 0,1; 0.1, .99,.....

%apply the weight to the binomial and get the an array of p-values
p1=zeros(1,length(a1));
outDis=[];
for i=1:length(a1)
   expectedDist                    =(normDistFrom_2LabeledSub*weightMat(i,1)+
normDistFrom_3LabeledSub*weightMat(i,2))*totalSteps;
   %expected distribution, a1*P([1 2 3],2,p)+a2*P([1 2 3],3,p) is a probability,
%     %which needs to multiply by the total number of bleaching steps, as chi square does
not work with ratio
   chi2stat = sum((obsDist-expectedDist).^2 ./ expectedDist); %get chi squares statistics,
manually
%      [~,p] = chi2gof([1 2 3],'freq',obsDist,'expected',expDist,'ctrs',[1 2 3],'nparams',1);
    p1(1,i)=1-chi2cdf(chi2stat,1) ;%get p value manually
%
    outDis=[outDis;[weightMat(i,:)*100,expectedDist/totalSteps]];
end
```

```
%find the maxima and the index of the maxima. This maxima is the height fit
%possibles
[~,In]=max(p1); %as we want to get maximum silimality
%get the distribution for the height p values

combDist                                =normDistFrom_2LabeledSub*weightMat(In,1)+
normDistFrom_3LabeledSub*weightMat(In,2);
%final expected ratio, it will be save as a results
expectedDist=combDist*totalSteps; %out of total steps, need to get chi square statistics
chi2stat = sum((obsDist-expectedDist).^2 ./ expectedDist);
%%%%%%%%%%%%%%%%%%%%%%%%----------------make the excel sheet to output
results ----------
fittingDecision=[{'Chi2GOF  decision'};{['p_Value=  ',num2str(p1(In)),'  chi2stat  =
',num2str(chi2stat),'  degree of freedom =2']}];
% percent of the each subunits
perentageOfSubunits=(weightMat(In,:)*100);
str=[{'% 2 Subunits'}, {'% 3 Subunits'}];
perentageOfSubunits=[str;num2cell(perentageOfSubunits)];

%make a header and arry to print results to plot with origin
header=[{'From  2  labeled  subunits'},{'From  3  labeled  subunits'},{'Expected
Ratio'},{'Observed Ratio'},{'Error in Observed Ratio'}];
finalDist=round([(normDistFrom_2LabeledSub*weightMat(In,1))',
(normDistFrom_3LabeledSub*weightMat(In,2))',combDist',...
   obsDistRatio',errorInObsDistRatio'],3);
finalDist=[header;num2cell(finalDist)];
column=[{''};{'1 step bleaching'};{'2 steps bleaching'};{'3 steps bleaching'}];
finalDist=[column,finalDist];


% Make a matrix to print the chi square statistics, decision
% of the hypothesis test
chi2TestResults=[{'Calclulated Chi Square value'};num2cell(chi2stat)];
resultsAndFitDecision=[perentageOfSubunits,[{''};{''}],[{''};{''}],chi2TestResults,fitting
Decision];

%make a matrix to print space
spacing=[[{''};{''}],[{''};{''}],[{''};{''}],[{''};{''}],[{''};{''}],[{''};{''}]];

%the final output matrix with r
Results=[spacing; resultsAndFitDecision;spacing;finalDist];

%make an matrix to print input data
inputDataHeader=[{'# 1 step'},{'# 2 step'},{'# 3 step'},{'probability'},{''},{''}];
inputData=[obsDist,prob];
```

```
inputData=[inputDataHeader;[num2cell(inputData),{''},{''}]];

%the final output matrix
outputExcel=[inputData;spacing;Results];

%get the saving file name and directory
if get(handles.saveAs,'Value')
   fileName=[get(handles.fileNameInput,'String'),'.xlsx'];
   filepath=pwd;
elseif get(handles.AutoGenerateFileName,'Value');
   fileName=['Stoicheometry fit with ', num2str(num_1steps), '-',num2str(num_2steps), '-
',num2str(num_3steps),'.xlsx'];
   filepath=pwd;
elseif  get(handles.showMeDirectoryAfterCalculation,'Value');
[fileName,filepath] = uiputfile('*.xlsx','Write a filename to save the data');
   if fileName==0
      return;
   end
end

if get(handles.displayResults,'Value')
    figure;
%     c = categorical({'one steps';'two steps';'three steps'});
   y = [obsDist',expectedDist'];
   bar(y)
   legend('Observed','Expected','Location','northwest','Orientation','horizontal');
   msgbox([{['Two labeled FP = ',num2str(perentageOfSubunits{2,1}),'% Three labeled
FP = ',...
      num2str(perentageOfSubunits{2,2}),'%']};{['p_Value= ',num2str(p1(In)),'  chi2stat =
',...
      num2str(chi2stat),'  degree of freedom =2']}]);
end

if get(handles.donotSaveResults,'Value')==0
   fullFileName=fullfile(filepath,fileName);
   xlswrite(fullFileName,outputExcel);
end

function probability_Callback(hObject, eventdata, handles)
% hObject    handle to probability (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of probability as text
%        str2double(get(hObject,'String')) returns contents of probability as a double
```

```
% --- Executes during object creation, after setting all properties.
function probability_CreateFcn(hObject, eventdata, handles)
% hObject    handle to probability (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in saveAs.
function saveAs_Callback(hObject, eventdata, handles)
% hObject    handle to saveAs (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of saveAs
if get(hObject,'Value')
   set(handles.showMeDirectoryAfterCalculation,'Value',0);
   set(handles.donotSaveResults,'Value',0);
   set(handles.fileNameInput,'Enable','on');
   set(handles.setFolder2,'Enable','on');

   set(handles.AutoGenerateFileName,'Value',0);
   set(handles.setFolder1,'Enable','off');

else
   set(handles.showMeDirectoryAfterCalculation,'Value',1);
   set(handles.setFolder1,'Enable','off');
   set(handles.saveAs,'Value',0);
   set(handles.fileNameInput,'Enable','off');
   set(handles.setFolder2,'Enable','off');
end
guidata(hObject, handles);

function fileNameInput_Callback(hObject, eventdata, handles)
% hObject    handle to fileNameInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of fileNameInput as text
%        str2double(get(hObject,'String')) returns contents of fileNameInput as a double
```

```
% --- Executes during object creation, after setting all properties.
function fileNameInput_CreateFcn(hObject, eventdata, handles)
% hObject    handle to fileNameInput (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in AutoGenerateFileName.
function AutoGenerateFileName_Callback(hObject, eventdata, handles)
% hObject    handle to AutoGenerateFileName (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of AutoGenerateFileName

if get(hObject,'Value')
   set(handles.setFolder2,'Enable','on');
   set(handles.showMeDirectoryAfterCalculation,'Value',0);
   set(handles.donotSaveResults,'Value',0);
   set(handles.saveAs,'Value',0);
   set(handles.fileNameInput,'Enable','off');
   set(handles.setFolder2,'Enable','off');
   set(handles.setFolder1,'Enable','on');
else
   set(handles.showMeDirectoryAfterCalculation,'Value',1);
   set(handles.setFolder1,'Enable','off');
end
guidata(hObject, handles);

% --- Executes on button press in showMeDirectoryAfterCalculation.
function showMeDirectoryAfterCalculation_Callback(hObject, eventdata, handles)
% hObject    handle to showMeDirectoryAfterCalculation (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of showMeDirectoryAfterCalculation
if get(hObject,'Value')
   set(handles.donotSaveResults,'Value',0);
   set(handles.saveAs,'Value',0);
   set(handles.fileNameInput,'Enable','off');
```

```
    set(handles.setFolder2,'Enable','off');

    set(handles.AutoGenerateFileName,'Value',0);
    set(handles.setFolder1,'Enable','off');

else
    set(handles.saveAs,'Value',1);
    set(handles.fileNameInput,'Enable','on');
    set(handles.setFolder2,'Enable','on');
end
guidata(hObject, handles);

% --- Executes on button press in donotSaveResults.
function donotSaveResults_Callback(hObject, eventdata, handles)
% hObject    handle to donotSaveResults (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of donotSaveResults
if get(hObject,'Value')
    set(handles.showMeDirectoryAfterCalculation,'Value',0);
    set(handles.saveAs,'Value',0);
    set(handles.fileNameInput,'Enable','off');
    set(handles.setFolder2,'Enable','off');

    set(handles.AutoGenerateFileName,'Value',0);
    set(handles.setFolder1,'Enable','off');

else
    set(handles.saveAs,'Value',1);
    set(handles.fileNameInput,'Enable','on');
    set(handles.setFolder2,'Enable','on');
end
guidata(hObject, handles);

% --- Executes on button press in setFolder2.
function setFolder2_Callback(hObject, eventdata, handles)
% hObject    handle to setFolder2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
currFolder=pwd;
addpath(currFolder)
folder_name = uigetdir;
cd(folder_name);

% --- Executes on button press in setFolder1.
```

```
function setFolder1_Callback(hObject, eventdata, handles)
% hObject    handle to setFolder1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
currFolder=pwd;
addpath(currFolder)
folder_name = uigetdir;
cd(folder_name);

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1 contents as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu1
contents = cellstr(get(handles.popupmenu1,'String')) ;
cond=contents{get(handles.popupmenu1,'Value')} ;
switch cond
   case 'Fit 2 and 3 labeled xFP'
      set(handles.num_3steps,'Visible','on');
      set(handles.text4,'Visible','on');
   case 'Fit 1 and 2 labeled xFP'
      set(handles.num_3steps,'Visible','off');
      set(handles.text4,'Visible','off');
end



guidata(hObject, handles);
% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if                ispc                &&                isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in displayResults.
function displayResults_Callback(hObject, eventdata, handles)
% hObject    handle to displayResults (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of displayResults
```

References

1. Colombo, M., Raposo, G., and Théry, C. (2014) Biogenesis, secretion, and intercellular interactions of exosomes and other extracellular vesicles. *Annual review of cell and developmental biology* **30**, 255-289

2. Johnsen, K. B., Gudbergsson, J. M., Skov, M. N., Pilgaard, L., Moos, T., and Duroux, M. (2014) A comprehensive overview of exosomes as drug delivery vehicles—endogenous nanocarriers for targeted cancer therapy. *Biochimica et Biophysica Acta (BBA)-Reviews on Cancer* **1846**, 75-87

3. H Rashed, M., Bayraktar, E., K Helal, G., Abd-Ellah, M. F., Amero, P., Chavez-Reyes, A., and Rodriguez-Aguayo, C. (2017) Exosomes: from garbage bins to promising therapeutic targets. *International journal of molecular sciences* **18**, 538

4. Corry, B., Rigby, P., Liu, Z.-W., and Martinac, B. (2005) Conformational changes involved in MscL channel gating measured using FRET spectroscopy. *Biophysical journal* **89**, L49-L51

5. Zhao, Y., Terry, D., Shi, L., Weinstein, H., Blanchard, S. C., and Javitch, J. A. (2010) Single-molecule dynamics of gating in a neurotransmitter transporter homologue. *Nature* **465**, 188

6. Wang, S., Vafabakhsh, R., Borschel, W. F., Ha, T., and Nichols, C. G. (2016) Structural dynamics of potassium-channel gating revealed by single-molecule FRET. *Nature Structural and Molecular Biology* **23**, 31

7. Schuler, B., and Eaton, W. A. (2008) Protein folding studied by single-molecule FRET. *Current opinion in structural biology* **18**, 16-26

8. Tan, Y.-W., Hanson, J. A., Chu, J.-W., and Yang, H. (2014) Confocal single-molecule FRET for protein conformational dynamics. in *Protein Dynamics*, Springer. pp 51-62

9. Zhang, Q., Zheng, X., Kuang, G., Wang, W., Zhu, L., Pang, R., Shi, X., Shang, X., Huang, X., Liu, P. N., and Lin, N. (2017) Single-Molecule Investigations of Conformation Adaptation of Porphyrins on Surfaces. *The Journal of Physical Chemistry Letters* **8**, 1241-1247

10. Richards, C. I., Luong, K., Srinivasan, R., Turner, S. W., Dougherty, D. A., Korlach, J., and Lester, H. A. (2012) Live-Cell Imaging of Single Receptor Composition Using Zero-Mode Waveguide Nanostructures. *Nano Letters* **12**, 3690-3694

11. Merchant, K. A., Best, R. B., Louis, J. M., Gopich, I. V., and Eaton, W. A. (2007) Characterizing the unfolded states of proteins using single-molecule FRET spectroscopy and molecular simulations. *Proceedings of the National Academy of Sciences* **104**, 1528-1533

12. Lipman, E. A., Schuler, B., Bakajin, O., and Eaton, W. A. (2003) Single-molecule measurement of protein folding kinetics. *Science* **301**, 1233-1235

13. Ulbrich, M. H., and Isacoff, E. Y. (2007) Subunit counting in membrane-bound proteins. *Nat Meth* **4**, 319-321

14. Harms, G. S., Cognet, L., Lommerse, P. H. M., Blab, G. A., Kahr, H., Gamsjäger, R., Spaink, H. P., Soldatov, N. M., Romanin, C., and Schmidt, T. (2001) Single-Molecule Imaging of L-Type Ca2+ Channels in Live Cells. *Biophysical Journal* **81**, 2639-2646

15. Aubin, J. (1979) Autofluorescence of viable cultured mammalian cells. *Journal of Histochemistry & Cytochemistry* **27**, 36-43

16. Andersson, H., Baechi, T., Hoechl, M., and Richter, C. (1998) Autofluorescence of living cells. *Journal of microscopy* **191**, 1-7

17. Stein, W. (2012) *The movement of molecules across cell membranes*, Elsevier

18. Liu, B., Mazouchi, A., and Gradinaru, C. C. (2010) Trapping Single Molecules in Liposomes: Surface Interactions and Freeze−Thaw Effects. *J. Phys. Chem. B* **114**, 15191-15198

19. Okumus, B., Wilson, T. J., Lilley, D. M., and Ha, T. (2004) Vesicle encapsulation studies reveal that single molecule ribozyme heterogeneities are intrinsic. *Biophysical journal* **87**, 2798-2806

20. Rosenkranz, T., Katranidis, A., Atta, D., Gregor, I., Enderlein, J., Grzelakowski, M., Rigler, P., Meier, W., and Fitter, J. (2009) Observing Proteins as Single Molecules Encapsulated in Surface-Tethered Polymeric Nanocontainers. *ChemBioChem* **10**, 702-709

21. Boukobza, E., Sonnenfeld, A., and Haran, G. (2001) Immobilization in surface-tethered lipid vesicles as a new tool for single biomolecule spectroscopy. *The Journal of Physical Chemistry B* **105**, 12165-12170

22. Anson, M. (1939) The denaturation of proteins by synthetic detergents and bile salts. *The Journal of general physiology* **23**, 239

23. Seddon, A. M., Curnow, P., and Booth, P. J. (2004) Membrane proteins, lipids and detergents: not just a soap opera. *Biochimica et Biophysica Acta (BBA)-Biomembranes* **1666**, 105-117

24. Sanderson, M. J., Smith, I., Parker, I., and Bootman, M. D. (2014) Fluorescence microscopy. *Cold Spring Harbor Protocols* **2014**, pdb. top071795

25. Martin, W. E., Srijanto, B. R., Collier, C. P., Vosch, T., and Richards, C. I. (2016) A Comparison of Single-Molecule Emission in Aluminum and Gold Zero-Mode Waveguides. *The Journal of Physical Chemistry A* **120**, 6719-6727

26. Murphy, D. B., and Davidson, M. W. (2012) Confocal laser scanning microscopy. *Fundamentals of Light Microscopy and Electronic Imaging, Second Edition*, 265-305

27. Funatsu, T., Harada, Y., Tokunaga, M., Saito, K., and Yanagida, T. (1995) Imaging of single fluorescent molecules and individual ATP turnovers by single myosin molecules in aqueous solution. *Nature* **374**, 555

28. Jain, A., Liu, R., Ramani, B., Arauz, E., Ishitsuka, Y., Ragunathan, K., Park, J., Chen, J., Xiang, Y. K., and Ha, T. (2011) Probing cellular protein complexes using single-molecule pull-down. *Nature* **473**, 484-488

29. Das, S. K., Liu, Y., Yeom, S., Kim, D. Y., and Richards, C. I. (2014) Single-Particle Fluorescence Intensity Fluctuations of Carbon Nanodots. *Nano Letters* **14**, 620-625

30. Fish, K. N. (2009) Total internal reflection fluorescence (TIRF) microscopy. *Current protocols in cytometry*, 12.18. 11-12.18. 13

31. Roy, R., Hohng, S., and Ha, T. (2008) A practical guide to single-molecule FRET. *Nature methods* **5**, 507

32. Zhang, Z., Baksh, M. M., Finn, M., Heidary, D. K., and Richards, C. I. (2016) Direct measurement of trafficking of the cystic fibrosis transmembrane conductance regulator to the cell surface and binding to a chemical chaperone. *Biochemistry* **56**, 240-249

33. Simpson, R. J. (2010) Disruption of cultured cells by nitrogen cavitation. *Cold Spring Harb Protoc* **2010**, pdb prot5513

34. Goldberg, S. (2008) Mechanical/physical methods of cell disruption and tissue homogenization. in *2D Page: Sample preparation and fractionation*, Springer. pp 3-22

35. Hernández-Zapata, E., Martínez-Balbuena, L., and Santamaría-Holek, I. (2009) Thermodynamics and dynamics of the formation of spherical lipid vesicles. *Journal of biological physics* **35**, 297-308

36. Markvoort, A., Pieterse, K., Steijaert, M., Spijker, P., and Hilbers, P. (2005) The Bilayer− Vesicle Transition Is Entropy Driven. *The Journal of Physical Chemistry B* **109**, 22649-22654

37. Martin, F. J., and Morano, J. K. (1988) Liposome extrusion method. Google Patents

38. Olson, F., Hunt, C., Szoka, F., Vail, W., and Papahadjopoulos, D. (1979) Preparation of liposomes of defined size distribution by extrusion through polycarbonate membranes. *Biochimica et Biophysica Acta (BBA)-Biomembranes* **557**, 9-23

39. Goh, W. J., Zou, S., Ong, W. Y., Torta, F., Alexandra, A. F., Schiffelers, R. M., Storm, G., Wang, J.-W., Czarny, B., and Pastorin, G. (2017) Bioinspired Cell-Derived Nanovesicles versus Exosomes as Drug Delivery Systems: a Cost-Effective Alternative. *Scientific reports* **7**, 14322

40. Jang, S. C., Kim, O. Y., Yoon, C. M., Choi, D.-S., Roh, T.-Y., Park, J., Nilsson, J., Lötvall, J., Kim, Y.-K., and Gho, Y. S. (2013) Bioinspired exosome-mimetic nanovesicles for targeted delivery of chemotherapeutics to malignant tumors. *ACS nano* **7**, 7698-7710

41. Clerc, S. G., and Thompson, T. E. (1994) A possible mechanism for vesicle formation by extrusion. *Biophysical journal* **67**, 475

42. Van Meer, G., Voelker, D. R., and Feigenson, G. W. (2008) Membrane lipids: where they are and how they behave. *Nature reviews Molecular cell biology* **9**, 112-124

43. Cooper, G. M. (2000) *The Cell: A Molecular Approach. 2nd edition*, Sinauer Associates, Sunderland (MA)

44. Li, X., and Donowitz, M. (2008) Fractionation of subcellular membrane vesicles of epithelial and nonepithelial cells by OptiPrep™ density gradient ultracentrifugation. *Exocytosis and Endocytosis*, 97-110

45. Graham, J., Ford, T., and Rickwood, D. (1994) The preparation of subcellular organelles from mouse liver in self-generated gradients of iodixanol. *Analytical biochemistry* **220**, 367-373

46. Simons, M., Krämer, E.-M., Thiele, C., Stoffel, W., and Trotter, J. (2000) Assembly of myelin by association of proteolipid protein with cholesterol-and galactosylceramide-rich membrane domains. *J Cell Biol* **151**, 143-154

47. Weerth, S. H., Holtzclaw, L. A., and Russell, J. T. (2007) Signaling proteins in raft-like microdomains are essential for Ca2+ wave propagation in glial cells. *Cell calcium* **41**, 155-167

48. Davies, M. J. (2004) Reactive species formed on proteins exposed to singlet oxygen. *Photochemical & photobiological sciences : Official journal of the European Photochemistry Association and the European Society for Photobiology* **3**, 17-25

49. Sies, H., and Menck, C. F. (1992) Singlet oxygen induced DNA damage. *Mutation research* **275**, 367-375

50. Jouonang, A. L., Didier, P., and Mely, Y. (2012) Identification of a thermally activated process in the Cy3 photobleaching mechanism. *Physical chemistry chemical physics : PCCP* **14**, 1585-1588

51. Aitken, C. E., Marshall, R. A., and Puglisi, J. D. (2008) An oxygen scavenging system for improvement of dye stability in single-molecule fluorescence experiments. *Biophysical journal* **94**, 1826-1835

52. Benesch, R. E., and Benesch, R. (1953) Enzymatic removal of oxygen for polarography and related methods. *Science* **118**, 447-448

53. Schwille, P., and Haustein, E. (2006) Fluorescence Correlation Spectroscopy - An Introduction to its Concepts and Applications.

54. Magde, D., Elson, E., and Webb, W. W. (1972) Thermodynamic Fluctuations in a Reacting System—Measurement by Fluorescence Correlation Spectroscopy. *Physical Review Letters* **29**, 705-708

55. Elson, E. L., and Magde, D. (1974) Fluorescence correlation spectroscopy. I. Conceptual basis and theory. *Biopolymers* **13**, 1-27

56. Magde, D. (1976) Chemical Kinetics and Fluorescence Correlation Spectroscopy. *Quarterly Reviews of Biophysics* **9**, 35-47

57. Itier, V., and Bertrand, D. (2001) Neuronal nicotinic receptors: from protein structure to function. *FEBS letters* **504**, 118-125

58. Millar, N. S., and Gotti, C. (2009) Diversity of vertebrate nicotinic acetylcholine receptors. *Neuropharmacology* **56**, 237-246

59. Jones, S., Sudweeks, S., and Yakel, J. L. (1999) Nicotinic receptors in the brain: correlating physiology with function. *Trends in neurosciences* **22**, 555-561

60. Frazier, C. J., Rollins, Y. D., Breese, C. R., Leonard, S., Freedman, R., and Dunwiddie, T. V. (1998) Acetylcholine activates an α-bungarotoxin-sensitive nicotinic current in rat hippocampal interneurons, but not pyramidal cells. *Journal of Neuroscience* **18**, 1187-1195

61. McGehee, D. S., Heath, M., Gelber, S., Devay, P., and Role, L. W. (1995) Nicotine enhancement of fast excitatory synaptic transmission in CNS by presynaptic receptors. *Science* **269**, 1692-1696

62. Wonnacott, S., Drasdo, A., Sanderson, E., and Rowell, P. (1990) Presynaptic nicotinic receptors and the modulation of transmitter release. in *The biology of nicotine dependence*, John Wiley and Sons, New York

63. Hillard, C. J. (1992) Nicotine-induced depolarization of cerebral cortical synaptosomes is dependent upon sodium. *Neuropharmacology* **31**, 909-914

64. Soliakov, L., Gallagher, T., and Wonnacott, S. (1995) Anatoxin-a-evoked [3H] dopamine release from rat striatal synaptosomes. *Neuropharmacology* **34**, 1535-1541

65. Gray, R., Rajan, A. S., Radcliffe, K. A., Yakehiro, M., and Dani, J. A. (1996) Hippocampal synaptic transmission enhanced by low concentrations of nicotine. *Nature* **383**, 713

66. Levin, E. D., and Simon, B. B. (1998) Nicotinic acetylcholine involvement in cognitive function in animals. *Psychopharmacology* **138**, 217-230

67. Rinne, J. O., Myllykyla¨, T., Lo¨nnberg, P., and Marjama¨ki, P. i. (1991) A postmortem study of brain nicotinic receptors in Parkinson's and Alzheimer's disease. *Brain Research* **547**, 155-158

68. Whitehouse, P. J., Martino, A., Wagster, M., Price, D., Mayeux, R., Atack, J., and Kellar, K. (1988) Reductions in [3H] nicotinic acetylcholine binding in Alzheimer's disease and Parkinson's disease An autobiographic study. *Neurology* **38**, 720-720

69. Quik, M., and Kulak, J. M. (2002) Nicotine and nicotinic receptors; relevance to Parkinson's disease. *Neurotoxicology* **23**, 581-594

70. Freedman, R., Hall, M., Adler, L. E., and Leonard, S. (1995) Evidence in postmortem brain tissue for decreased numbers of hippocampal nicotinic receptors in schizophrenia. *Biological psychiatry* **38**, 22-33

71. Adler, L. E., Olincy, A., Waldo, M., Harris, J. G., Griffith, J., Stevens, K., Flach, K., Nagamoto, H., Bickford, P., and Leonard, S. (1998) Schizophrenia, sensory gating, and nicotinic receptors. *Schizophrenia bulletin* **24**, 189

72. Jin, Y., Huang, X., Papke, R. L., Jutkiewicz, E. M., Showalter, H. D., and Zhan, C.-G. (2017) Design, synthesis, and biological activity of 5′-phenyl-1,2,5,6-tetrahydro-3,3′-bipyridine analogues as potential antagonists of nicotinic acetylcholine receptors. *Bioorganic & Medicinal Chemistry Letters* **27**, 4350-4353

73. Freedman, R., Hall, M., Adler, L. E., and Leonard, S. Evidence in postmortem brain tissue for decreased numbers of hippocampal nicotinic receptors in schizophrenia. *Biological Psychiatry* **38**, 22-33

74. Albuquerque, E. X., Pereira, E. F., Alkondon, M., and Rogers, S. W. (2009) Mammalian nicotinic acetylcholine receptors: from structure to function. *Physiological reviews* **89**, 73-120

75. Morales-Perez, C. L., Noviello, C. M., and Hibbs, R. E. (2016) X-ray structure of the human α4β2 nicotinic receptor. *Nature* **538**, 411-415

76. Changeux, J.-P. (2010) Nicotine addiction and nicotinic receptors: lessons from genetically modified mice. *Nat Rev Neurosci* **11**, 389-401

77. Huang, X., Zheng, F., Crooks, P. A., Dwoskin, L. P., and Zhan, C.-G. (2005) Modeling multiple species of nicotine and deschloroepibatidine interacting with α4β2 nicotinic acetylcholine receptor: from microscopic binding to phenomenological binding affinity. *Journal of the American Chemical Society* **127**, 14401-14414

78. Huang, X., Zheng, F., Stokes, C., Papke, R. L., and Zhan, C.-G. (2008) Modeling binding modes of α7 nicotinic acetylcholine receptor with ligands: the roles of Gln117 and other residues of the receptor in agonist binding. *Journal of medicinal chemistry* **51**, 6293-6302

79. Huang, X., Zheng, F., Chen, X., Crooks, P. A., Dwoskin, L. P., and Zhan, C.-G. (2006) Modeling subtype-selective agonists binding with α4β2 and α7 nicotinic acetylcholine receptors: effects of local binding and long-range electrostatic interactions. *Journal of medicinal chemistry* **49**, 7661-7674

80. Taly, A., Corringer, P.-J., Guedin, D., Lestage, P., and Changeux, J.-P. (2009) Nicotinic receptors: allosteric transitions and therapeutic targets in the nervous system. *Nature Reviews Drug Discovery* **8**, 733

81. Melroy-Greif, W. E., Stitzel, J. A., and Ehringer, M. A. (2016) Nicotinic acetylcholine receptors: upregulation, age-related effects and associations with drug use. *Genes, Brain and Behavior* **15**, 89-107

82. Albuquerque, E. X., Pereira, E. F. R., Alkondon, M., and Rogers, S. W. (2009) Mammalian nicotinic acetycholine receptors: from structure to function. *Physiol. Rev.* **89**, 73-120

83. Changeux, J.-P., and Edelstein, S. J. (1998) Allosteric receptors after 30 years. *Neuron* **21**, 959-980

84. Gotti, C., and Clementi, F. (2004) Neuronal nicotinic receptors: from structure to pathology. *Progress in neurobiology* **74**, 363-396

85. Dau, A., Komal, P., Truong, M., Morris, G., Evans, G., and Nashmi, R. (2013) RIC-3 differentially modulates α4β2 and α7 nicotinic receptor assembly, expression, and nicotine-induced receptor upregulation. *BMC neuroscience* **14**, 47

86. Lester, H. A., Xiao, C., Srinivasan, R., Son, C. D., Miwa, J., Pantoja, R., Banghart, M. R., Dougherty, D. A., Goate, A. M., and Wang, J. C. (2009) Nicotine is a selective pharmacological chaperone of acetylcholine receptor number and stoichiometry. Implications for drug discovery. *The AAPS journal* **11**, 167-177

87. Tapia, L., Kuryatov, A., and Lindstrom, J. (2007) Ca2+ permeability of the (α4) 3 (β2) 2 stoichiometry greatly exceeds that of (α4) 2 (β2) 3 human acetylcholine receptors. *Molecular pharmacology* **71**, 769-776

88. Lax, P., Fucile, S., and Eusebi, F. (2002) Ca2+ permeability of human heteromeric nAChRs expressed by transfection in human cells. *Cell calcium* **32**, 53-58

89. Moroni, M., and Bermudez, I. (2006) Stoichiometry and pharmacology of two human α4β2 nicotinic receptor types. *Journal of Molecular Neuroscience* **30**, 95-96

90. Exley, R., Moroni, M., Sasdelli, F., Houlihan, L. M., Lukas, R. J., Sher, E., Zwart, R., and Bermudez, I. (2006) Chaperone protein 14-3-3 and protein kinase A increase the relative abundance of low agonist sensitivity human α4β2 nicotinic acetylcholine receptors in Xenopus oocytes. *Journal of neurochemistry* **98**, 876-885

91. Picciotto, M. R., Zoli, M., Rimondini, R., Léna, C., Marubio, L. M., Pich, E. M., Fuxe, K., and Changeux, J.-P. (1998) Acetylcholine receptors containing the β2 subunit are involved in the reinforcing properties of nicotine. *Nature* **391**, 173

92. Marubio, L., Gardier, A., Durier, S., David, D., Klink, R., Arroyo-Jimenez, M., McIntosh, J., Rossi, F., Champtiaux, N., and Zoli, M. (2003) Effects of nicotine in the dopaminergic system of mice lacking the alpha4 subunit of neuronal nicotinic acetylcholine receptors. *European Journal of Neuroscience* **17**, 1329-1337

93. Sziraki, I., Lipovac, M., Hashim, A., Sershen, H., Allen, D., Cooper, T., Czobor, P., and Lajtha, A. (2001) Differences in nicotine-induced dopamine release and nicotine pharmacokinetics between Lewis and Fischer 344 rats. *Neurochemical research* **26**, 609-617

94. Tapper, A. R., McKinney, S. L., Nashmi, R., Schwarz, J., Deshpande, P., Labarca, C., Whiteaker, P., Marks, M. J., Collins, A. C., and Lester, H. A. (2004) Nicotine activation of α4* receptors: sufficient for reward, tolerance, and sensitization. *Science* **306**, 1029-1032

95. Drisdel, R. C., and Green, W. N. (2000) Neuronal α-bungarotoxin receptors are α7 subunit homomers. *Journal of Neuroscience* **20**, 133-139

96. Anand, R., Peng, X., and Lindstrom, J. (1993) Homomeric and native α7 acetylcholine receptors exhibit remarkably similar but non-identical pharmacological properties, suggesting that the native receptor is a heteromeric protein complex. *FEBS letters* **327**, 241-246

97. Orr-Urtreger, A., Göldner, F. M., Saeki, M., Lorenzo, I., Goldberg, L., De Biasi, M., Dani, J. A., Patrick, J. W., and Beaudet, A. L. (1997) Mice deficient in the α7 neuronal nicotinic acetylcholine receptor lack α-bungarotoxin binding sites and hippocampal fast nicotinic currents. *Journal of Neuroscience* **17**, 9165-9171

98. Chen, D., and Patrick, J. W. (1997) The α-bungarotoxin-binding nicotinic acetylcholine receptor from rat brain contains only the α7 subunit. *Journal of Biological Chemistry* **272**, 24024-24029

99. Barrantes, G. E., Rogers, A. T., Lindstrom, J., and Wonnacott, S. (1995) α-Bungarotoxin binding sites in rat hippocampal and cortical cultures: initial characterisation, colocalisation with α7 subunits and up-regulation by chronic nicotine treatment. *Brain research* **672**, 228-236

100. Suzuki, T., Hide, I., Matsubara, A., Hama, C., Harada, K., Miyano, K., Andrä, M., Matsubayashi, H., Sakai, N., and Kohsaka, S. (2006) Microglial α7 nicotinic acetylcholine receptors drive a phospholipase C/IP3 pathway and modulate the cell activation toward a neuroprotective role. *Journal of neuroscience research* **83**, 1461-1470

101. Peng, X., Gerzanich, V., Anand, R., Wang, F., and Lindstrom, J. (1997) Chronic nicotine treatment up-regulates α3 and α7 acetylcholine receptor subtypes expressed by the human neuroblastoma cell line SH-SY5Y. *Molecular pharmacology* **51**, 776-784

102. Liu, Q., Huang, Y., Xue, F., Simard, A., DeChon, J., Li, G., Zhang, J., Lucero, L., Wang, M., and Sierks, M. (2009) A novel nicotinic acetylcholine receptor subtype in basal forebrain cholinergic neurons with high sensitivity to amyloid peptides. *Journal of Neuroscience* **29**, 918-929

103. Liu, Q., Huang, Y., Shen, J., Steffensen, S., and Wu, J. (2012) Functional α7β2 nicotinic acetylcholine receptors expressed in hippocampal interneurons exhibit high sensitivity to pathological level of amyloid β peptides. *BMC neuroscience* **13**, 155

104. Nielsen, B. E., Minguez, T., Bermudez, I., and Bouzat, C. (2018) Molecular function of the novel α7β2 nicotinic receptor. *Cellular and Molecular Life Sciences*, 1-15

105. Séguéla, P., Wadiche, J., Dineley-Miller, K., Dani, J. A., and Patrick, J. W. (1993) Molecular cloning, functional properties, and distribution of rat brain alpha 7: a nicotinic cation channel highly permeable to calcium. *Journal of Neuroscience* **13**, 596-604

106. Stoker, A. K., Olivier, B., and Markou, A. (2012) Role of α7- and β4-Containing Nicotinic Acetylcholine Receptors in the Affective and Somatic Aspects of Nicotine Withdrawal: Studies in Knockout Mice. *Behavior Genetics* **42**, 423-436

107. Mazzo, F., Pistillo, F., Grazioso, G., Clementi, F., Borgese, N., Gotti, C., and Colombo, S. F. (2013) Nicotine-modulated subunit stoichiometry affects stability and trafficking of α3β4 nicotinic receptor. *Journal of Neuroscience* **33**, 12316-12328

108. Russell, M., Jarvis, M., Iyer, R., and Feyerabend, C. (1980) Relation of nicotine yield of cigarettes to blood nicotine concentrations in smokers. *Br Med J* **280**, 972-976

109. Wall, T. R., Henderson, B. J., Voren, G., Wageman, C. R., Deshpande, P., Cohen, B. N., Grady, S. R., Marks, M. J., Yohannes, D., and Kenny, P. J. (2017) TC299423, a Novel Agonist for Nicotinic Acetylcholine Receptors. *Frontiers in pharmacology* **8**, 641

110. Albillos, A., and McIntosh, J. M. (2018) Human nicotinic receptors in chromaffin cells: characterization and pharmacology. *Pflügers Archiv-European Journal of Physiology* **470**, 21-27

111. Yuan, M., Malagon, A. M., Yasuda, D., Belluzzi, J. D., Leslie, F. M., and Zaveri, N. T. (2017) The α3β4 nAChR partial agonist AT-1001 attenuates stress-induced reinstatement of nicotine seeking in a rat model of relapse and induces minimal withdrawal in dependent rats. *Behavioural brain research* **333**, 251-257

112. Sarasamkan, J., Fischer, S., Deuther-Conrad, W., Ludwig, F.-A., Scheunemann, M., Arunrungvichian, K., Vajragupta, O., and Brust, P. (2017) Radiosynthesis of (S)-[18F] T1: The first PET radioligand for molecular imaging of α3β4 nicotinic acetylcholine receptors. *Applied Radiation and Isotopes* **124**, 106-113

113. de Moura, F. B., and McMahon, L. R. (2017) The contribution of α4β2 and non-α4β2 nicotinic acetylcholine receptors to the discriminative stimulus effects of nicotine and varenicline in mice. *Psychopharmacology* **234**, 781-792

114. Salas, R., Pieri, F., and De Biasi, M. (2004) Decreased signs of nicotine withdrawal in mice null for the β4 nicotinic acetylcholine receptor subunit. *Journal of Neuroscience* **24**, 10035-10039

115. Jackson, K. J., Sanjakdar, S. S., Muldoon, P. P., McIntosh, J. M., and Damaj, M. I. (2013) The α3β4* nicotinic acetylcholine receptor subtype mediates nicotine reward and physical nicotine withdrawal signs independently of the α5 subunit in the mouse. *Neuropharmacology* **70**, 228-235

116. Creese, I., and Sibley, D. R. (1981) Receptor adaptations to centrally acting drugs. *Annual Review of Pharmacology and Toxicology* **21**, 357-391

117. Wonnacott, S. (1990) The paradox of nicotinic acetylcholine receptor upregulation by nicotine. *Trends in pharmacological sciences* **11**, 216-219

118. Henderson, B. J., and Lester, H. A. (2015) Inside-out neuropharmacology of nicotinic drugs. *Neuropharmacology* **96**, 178-193

119. Benwell, M. E., Balfour, D. J., and Anderson, J. (1988) Evidence that tobacco smoking increases the density of (−)-[3H] nicotine binding sites in human brain. *Journal of neurochemistry* **50**, 1243-1247

120. Schwartz, R. D., and Kellar, K. J. (1983) Nicotinic cholinergic receptor binding sites in the brain: regulation in vivo. *Science* **220**, 214-216

121. Marks, M. J., Burch, J. B., and Collins, A. C. (1983) Effects of chronic nicotine infusion on tolerance development and nicotinic receptors. *Journal of Pharmacology and Experimental Therapeutics* **226**, 817-825

122. Pauly, J., Marks, M., Gross, S., and Collins, A. (1991) An autoradiographic analysis of cholinergic receptors in mouse brain after chronic nicotine treatment. *Journal of Pharmacology and Experimental Therapeutics* **258**, 1127-1136

123. Gopalakrishnan, M., Molinari, E. J., and Sullivan, J. P. (1997) Regulation of human α4β2 neuronal nicotinic acetylcholine receptors by cholinergic channel ligands and second messenger pathways. *Molecular pharmacology* **52**, 524-534

124. Gaimarri, A., Moretti, M., Riganti, L., Zanardi, A., Clementi, F., and Gotti, C. (2007) Regulation of neuronal nicotinic receptor traffic and expression. *Brain research reviews* **55**, 134-143

125. Zatonski, W., Cedzynska, M., Tutka, P., and West, R. (2006) An uncontrolled trial of cytisine (Tabex) for smoking cessation. *Tobacco Control* **15**, 481-484

126. Rollema, H., Coe, J. W., Chambers, L. K., Hurst, R. S., Stahl, S. M., and Williams, K. E. (2007) Rationale, pharmacology and clinical efficacy of partial agonists of α4β2 nACh receptors for smoking cessation. *Trends in pharmacological sciences* **28**, 316-325

127. Mihalak, K. B., Carroll, F. I., and Luetje, C. W. (2006) Varenicline is a partial agonist at α4β2 and a full agonist at α7 neuronal nicotinic receptors. *Molecular pharmacology* **70**, 801-805

128. Turner, J. R., Castellano, L. M., and Blendy, J. A. (2010) Parallel anxiolytic-like effects and upregulation of neuronal nicotinic acetylcholine receptors following chronic nicotine and varenicline. *Nicotine & Tobacco Research* **13**, 41-46

129. Wilkes, S. (2008) The use of bupropion SR in cigarette smoking cessation. *International journal of chronic obstructive pulmonary disease* **3**, 45

130. Fava, M., Rush, A. J., Thase, M. E., Clayton, A., Stahl, S. M., Pradko, J. F., and Johnston, J. A. (2005) 15 years of clinical experience with bupropion HCl: from bupropion to bupropion SR to bupropion XL. *Primary care companion to the Journal of clinical psychiatry* **7**, 106

131. Boyd, N. D. (1987) Two distinct kinetic phases of desensitization of acetylcholine receptors of clonal rat PC12 cells. *The Journal of Physiology* **389**, 45-67

132. Fenster, C. P., Whitworth, T. L., Sheffield, E. B., Quick, M. W., and Lester, R. A. (1999) Upregulation of surface α4β2 nicotinic receptors is initiated by receptor desensitization after chronic exposure to nicotine. *Journal of Neuroscience* **19**, 4804-4814

133. Kishi, M., and Steinbach, J. H. (2006) Role of the agonist binding site in up-regulation of neuronal nicotinic α4β2 receptors. *Molecular pharmacology* **70**, 2037-2044

134. Kuryatov, A., Luo, J., Cooper, J., and Lindstrom, J. (2005) Nicotine acts as a pharmacological chaperone to up-regulate human α4β2 acetylcholine receptors. *Molecular Pharmacology* **68**, 1839-1851

135. Vallejo, Y. F., Buisson, B., Bertrand, D., and Green, W. N. (2005) Chronic nicotine exposure upregulates nicotinic receptors by a novel mechanism. *Journal of Neuroscience* **25**, 5563-5572

136. Darsow, T., Booker, T., Piña-Crespo, J. C., and Heinemann, S. F. (2005) Exocytic trafficking is required for nicotine-induced up-regulation of α4β2 nicotinic acetylcholine receptors. *Journal of Biological Chemistry* **280**, 18311-18320

137. Sallette, J., Bohler, S., Benoit, P., Soudant, M., Pons, S., Le Novere, N., Changeux, J.-P., and Corringer, P. J. (2004) An extracellular protein microdomain controls up-regulation of neuronal nicotinic acetylcholine receptors by nicotine. *Journal of Biological Chemistry* **279**, 18767-18775

138. Sallette, J., Pons, S., Devillers-Thiery, A., Soudant, M., Prado de Carvalho, L., Changeux, J.-P., and Corringer, P. J. (2005) Nicotine Upregulates Its Own Receptors through Enhanced Intracellular Maturation. *Neuron* **46**, 595-607

139. Srinivasan, R., Pantoja, R., Moss, F. J., Mackey, E. D., Son, C. D., Miwa, J., and Lester, H. A. (2011) Nicotine up-regulates α4β2 nicotinic receptors and ER exit sites via stoichiometry-dependent chaperoning. *The Journal of general physiology* **137**, 59-79

140. Henderson, B. J., Srinivasan, R., Nichols, W. A., Dilworth, C. N., Gutierrez, D. F., Mackey, E. D. W., McKinney, S., Drenan, R. M., Richards, C. I., and Lester, H. A. (2014) Nicotine exploits a COPI-mediated process for chaperone-mediated up-regulation of its receptors. *Journal of General Physiology* **143**, 51-66

141. Richards, C. I., Srinivasan, R., Xiao, C., Mackey, E. D. W., Miwa, J. M., and Lester, H. A. (2011) Trafficking of α4* Nicotinic Receptors Revealed by Superecliptic Phluorin. *J. Biol. Chem.* **286**, 31241-31249

142. Peng, X., Gerzanich, V., Anand, R., Whiting, P. J., and Lindstrom, J. (1994) Nicotine-induced increase in neuronal nicotinic receptors results from a decrease in the rate of receptor turnover. *Molecular pharmacology* **46**, 523-530

143. Sajid, M. I., Jamshaid, U., Jamshaid, T., Zafar, N., Fessi, H., and Elaissari, A. (2016) Carbon nanotubes from synthesis to in vivo biomedical applications. *International journal of pharmaceutics* **501**, 278-299

144. Bianco, A., Kostarelos, K., and Prato, M. (2005) Applications of carbon nanotubes in drug delivery. *Current opinion in chemical biology* **9**, 674-679

145. Liu, Z., Tabakman, S., Welsher, K., and Dai, H. (2009) Carbon nanotubes in biology and medicine: in vitro and in vivo detection, imaging and drug delivery. *Nano research* **2**, 85-120

146. Mezei, M., and Gulasekharam, V. (1980) Liposomes-a selective drug delivery system for the topical route of administration I. Lotion dosage form. *Life Sciences* **26**, 1473-1477

147. Allen, T. M., and Cullis, P. R. (2013) Liposomal drug delivery systems: from concept to clinical applications. *Advanced drug delivery reviews* **65**, 36-48

148. Nasongkla, N., Bey, E., Ren, J., Ai, H., Khemtong, C., Guthi, J. S., Chin, S.-F., Sherry, A. D., Boothman, D. A., and Gao, J. (2006) Multifunctional polymeric micelles as cancer-targeted, MRI-ultrasensitive drug delivery systems. *Nano letters* **6**, 2427-2430

149. Hrubý, M., Koňák, Č., and Ulbrich, K. (2005) Polymeric micellar pH-sensitive drug delivery system for doxorubicin. *Journal of Controlled Release* **103**, 137-148

150. Soppimath, K. S., Aminabhavi, T. M., Kulkarni, A. R., and Rudzinski, W. E. (2001) Biodegradable polymeric nanoparticles as drug delivery devices. *Journal of controlled release* **70**, 1-20

151. Sercombe, L., Veerati, T., Moheimani, F., Wu, S. Y., Sood, A. K., and Hua, S. (2015) Advances and challenges of liposome assisted drug delivery. *Frontiers in pharmacology* **6**, 286

152. Bulbake, U., Doppalapudi, S., Kommineni, N., and Khan, W. (2017) Liposomal formulations in clinical use: an updated review. *Pharmaceutics* **9**, 12

153. Lopez-Pinto, J., Gonzalez-Rodriguez, M., and Rabasco, A. (2005) Effect of cholesterol and ethanol on dermal delivery from DPPC liposomes. *International journal of pharmaceutics* **298**, 1-12

154. Kirby, C., Clarke, J., and Gregoriadis, G. (1980) Effect of the cholesterol content of small unilamellar liposomes on their stability in vivo and in vitro. *Biochemical Journal* **186**, 591

155. Koning, G. A., and Storm, G. (2003) Targeted drug delivery systems for the intracellular delivery of macromolecular drugs. *Drug discovery today* **8**, 482-483

156. Moonschi, F. H., Hughes, C. B., Mussman, G. M., Fowlkes, J. L., Richards, C. I., and Popescu, I. (2017) Advances in micro- and nanotechnologies for the GLP-1-based therapy and imaging of pancreatic beta-cells. *Acta Diabetologica*

157. Pattni, B. S., Chupin, V. V., and Torchilin, V. P. (2015) New developments in liposomal drug delivery. *Chemical reviews* **115**, 10938-10966

158. Ulrich, A. S. (2002) Biophysical aspects of using liposomes as delivery vehicles. *Bioscience reports* **22**, 129-150

159. Campbell, R. B., Fukumura, D., Brown, E. B., Mazzola, L. M., Izumi, Y., Jain, R. K., Torchilin, V. P., and Munn, L. L. (2002) Cationic charge determines the distribution of liposomes between the vascular and extravascular compartments of tumors. *Cancer research* **62**, 6831-6836

160. Johnson, S. (1973) The effect of charge and cholesterol on the size and thickness of sonicated phospholipid vesicles. *Biochimica et Biophysica Acta (BBA)-Biomembranes* **307**, 27-41

161. Bangham, A. D., De Gier, J., and Greville, G. (1967) Osmotic properties and water permeability of phospholipid liquid crystals. *Chemistry and physics of lipids* **1**, 225-246

162. Parente, R. A., and Lentz, B. R. (1984) Phase behavior of large unilamellar vesicles composed of synthetic phospholipids. *Biochemistry* **23**, 2353-2362

163. Hargreaves, W. R., and Deamer, D. W. (1978) Liposomes from ionic, single-chain amphiphiles. *Biochemistry* **17**, 3759-3768

164. Biswas, S., Dodwadkar, N. S., Deshpande, P. P., and Torchilin, V. P. (2012) Liposomes loaded with paclitaxel and modified with novel triphenylphosphonium-PEG-PE conjugate possess low toxicity, target mitochondria and demonstrate enhanced antitumor effects in vitro and in vivo. *Journal of controlled release* **159**, 393-402

165. Hope, M., Bally, M., Webb, G., and Cullis, P. (1985) Production of large unilamellar vesicles by a rapid extrusion procedure. Characterization of size distribution, trapped volume and ability to maintain a membrane potential. *Biochimica et Biophysica Acta (BBA)-Biomembranes* **812**, 55-65

166. Qin, J., and Xu, Q. (2014) Functions and application of exosomes. *Acta Pol Pharm* **71**, 537-543

167. Harding, C., Heuser, J., and Stahl, P. (1983) Receptor-mediated endocytosis of transferrin and recycling of the transferrin receptor in rat reticulocytes. *The Journal of cell biology* **97**, 329-339

168. Pan, B.-T., and Johnstone, R. M. (1983) Fate of the transferrin receptor during maturation of sheep reticulocytes in vitro: selective externalization of the receptor. *Cell* **33**, 967-978

169. Urbanelli, L., Magini, A., Buratta, S., Brozzi, A., Sagini, K., Polchi, A., Tancini, B., and Emiliani, C. (2013) Signaling pathways in exosomes biogenesis, secretion and fate. *Genes* **4**, 152-170

170. Raposo, G., and Stoorvogel, W. (2013) Extracellular vesicles: exosomes, microvesicles, and friends. *J Cell Biol* **200**, 373-383

171. Kooijmans, S. A., Vader, P., van Dommelen, S. M., van Solinge, W. W., and Schiffelers, R. M. (2012) Exosome mimetics: a novel class of drug delivery systems. *International journal of nanomedicine* **7**, 1525

172. Ha, D., Yang, N., and Nadithe, V. (2016) Exosomes as therapeutic drug carriers and delivery vehicles across biological membranes: current perspectives and future challenges. *Acta Pharmaceutica Sinica B* **6**, 287-296

173. Tian, Y., Li, S., Song, J., Ji, T., Zhu, M., Anderson, G. J., Wei, J., and Nie, G. (2014) A doxorubicin delivery platform using engineered natural membrane vesicle exosomes for targeted tumor therapy. *Biomaterials* **35**, 2383-2390

174. Yang, T., Martin, P., Fogarty, B., Brown, A., Schurman, K., Phipps, R., Yin, V. P., Lockman, P., and Bai, S. (2015) Exosome delivered anticancer drugs across the blood-brain barrier for brain cancer therapy in Danio rerio. *Pharmaceutical research* **32**, 2003-2014

175. Jevševar, S., Kunstelj, M., and Porekar, V. G. (2010) PEGylation of therapeutic proteins. *Biotechnology journal* **5**, 113-128

176. Harris, J. M., and Chess, R. B. (2003) Effect of pegylation on pharmaceuticals. *Nature reviews. Drug discovery* **2**, 214

177. Baker, D. P., Lin, E. Y., Lin, K., Pellegrini, M., Petter, R. C., Chen, L. L., Arduini, R. M., Brickelmaier, M., Wen, D., and Hess, D. M. (2006) N-Terminally PEGylated Human Interferon-β-1a with Improved Pharmacokinetic Properties and in Vivo Efficacy in a Melanoma Angiogenesis Model §. *Bioconjugate chemistry* **17**, 179-188

178. Danhier, F., Ansorena, E., Silva, J. M., Coco, R., Le Breton, A., and Préat, V. (2012) PLGA-based nanoparticles: an overview of biomedical applications. *Journal of controlled release* **161**, 505-522

179. Hans, M., and Lowman, A. (2002) Biodegradable nanoparticles for drug delivery and targeting. *Current Opinion in Solid State and Materials Science* **6**, 319-327

180. Betancourt, T., Byrne, J. D., Sunaryo, N., Crowder, S. W., Kadapakkam, M., Patel, S., Casciato, S., and Brannon-Peppas, L. (2009) PEGylation strategies for active targeting of PLA/PLGA nanoparticles. *Journal of biomedical materials research Part A* **91**, 263-276

181. Araújo, F., Shrestha, N., Shahbazi, M.-A., Fonte, P., Mäkilä, E. M., Salonen, J. J., Hirvonen, J. T., Granja, P. L., Santos, H. A., and Sarmento, B. (2014) The impact of nanoparticles on the mucosal translocation and transport of GLP-1 across the intestinal epithelium. *Biomaterials* **35**, 9199-9207

182. Araújo, F., Shrestha, N., Gomes, M., Herranz-Blanco, B., Liu, D., Hirvonen, J., Granja, P., Santos, H., and Sarmento, B. (2016) In vivo dual-delivery of glucagon like peptide-1 (GLP-1) and dipeptidyl peptidase-4 (DPP4) inhibitor through composites prepared by microfluidics for diabetes therapy. *Nanoscale* **8**, 10706

183. Joseph, J., Kalitsky, J., St-Pierre, S., and Brubaker, P. (2000) Oral delivery of glucagon-like peptide-1 in a modified polymer preparation normalizes basal glycaemia in diabetic db/db mice. *Diabetologia* **43**, 1319-1328

184. Gao, Z., Tang, Y., Chen, J., Bai, R., Zhang, Q., Hou, Y., Lu, Y., and Bai, G. (2009) A novel DPP-IV-resistant analog of glucagon-like peptide-1 (GLP-1): KGLP-1 alone or in combination with long-acting PLGA microspheres. *Peptides* **30**, 1874-1881

185. Yin, D., Lu, Y., Zhang, H., Zhang, G., Zou, H., Sun, D., and Zhong, Y. (2008) Preparation of glucagon-like peptide-1 loaded PLGA microspheres: characterizations, release studies and bioactivities in vitro/in vivo. *Chemical and Pharmaceutical Bulletin* **56**, 156-161

186. Grenha, A., Grainger, C. I., Dailey, L. A., Seijo, B., Martin, G. P., Remuñán-López, C., and Forbes, B. (2007) Chitosan nanoparticles are compatible with respiratory epithelial cells in vitro. *european journal of pharmaceutical sciences* **31**, 73-84

187. Zhang, W. F., Zhao, X. T., Zhao, Q. S., Zha, S. H., Liu, D. M., Zheng, Z. J., Li, W. T., Zhou, H. Y., and Yan, F. (2014) Biocompatibility and characteristics of

theophylline/carboxymethyl chitosan microspheres for pulmonary drug delivery. *Polymer International* **63**, 1035-1040

188. Verheul, R. J., Amidi, M., van Steenbergen, M. J., van Riet, E., Jiskoot, W., and Hennink, W. E. (2009) Influence of the degree of acetylation on the enzymatic degradation and in vitro biological properties of trimethylated chitosans. *Biomaterials* **30**, 3129-3135

189. Poth, N., Seiffart, V., Gross, G., Menzel, H., and Dempwolf, W. (2015) Biodegradable chitosan nanoparticle coatings on titanium for the delivery of BMP-2. *Biomolecules* **5**, 3-19

190. Yamamoto, H., Kuno, Y., Sugimoto, S., Takeuchi, H., and Kawashima, Y. (2005) Surface-modified PLGA nanosphere with chitosan improved pulmonary delivery of calcitonin by mucoadhesion and opening of the intercellular tight junctions. *Journal of controlled Release* **102**, 373-381

191. Lehr, C.-M., Bouwstra, J. A., Schacht, E. H., and Junginger, H. E. (1992) In vitro evaluation of mucoadhesive properties of chitosan and some other natural polymers. *International journal of Pharmaceutics* **78**, 43-48

192. Ariful Islam, M., Park, T.-E., Reesor, E., Cherukula, K., Hasan, A., Firdous, J., Singh, B., Kang, S.-K., Choi, Y.-J., and Park, I.-K. (2015) Mucoadhesive chitosan derivatives as novel drug carriers. *Current pharmaceutical design* **21**, 4285-4309

193. Ambrozik, J., Zimoch, A., Jarmoluk, A., and Semeriak, K. (2011) Enzymatic degradation of chitosan with lysozyme or cellulase. *Przemysl Chemiczny* **90**, 676-679

194. Wang, J. J., Zeng, Z. W., Xiao, R. Z., Xie, T., Zhou, G. L., Zhan, X. R., and Wang, S. L. (2011) Recent advances of chitosan nanoparticles as drug carriers. *International journal of nanomedicine* **6**, 765

195. Ishida, T., Wang, X., Shimizu, T., Nawata, K., and Kiwada, H. (2007) PEGylated liposomes elicit an anti-PEG IgM response in a T cell-independent manner. *Journal of Controlled Release* **122**, 349-355

196. Kaasalainen, M., Rytkönen, J., Mäkilä, E., Närvänen, A., and Salonen, J. (2015) Electrostatic interaction on loading of therapeutic peptide GLP-1 into porous silicon nanoparticles. *Langmuir* **31**, 1722-1729

197. Huotari, A., Xu, W., Mönkäre, J., Kovalainen, M., Herzig, K.-H., Lehto, V.-P., and Järvinen, K. (2013) Effect of surface chemistry of porous silicon microparticles on glucagon-like peptide-1 (GLP-1) loading, release and biological activity. *International journal of pharmaceutics* **454**, 67-73

198. Shrestha, N., Araújo, F., Shahbazi, M.-A., Mäkilä, E., Gomes, M. J., Airavaara, M., Kauppinen, E. I., Raula, J., Salonen, J., and Hirvonen, J. (2016) Oral hypoglycaemic effect of GLP-1 and DPP4 inhibitor based nanocomposites in a diabetic animal model. *Journal of Controlled Release* **232**, 113-119

199. Fuhrmann, G., Herrmann, I. K., and Stevens, M. M. (2015) Cell-derived vesicles for drug therapy and diagnostics: opportunities and challenges. *Nano Today* **10**, 397-409

200. Gercel-Taylor, C., Atay, S., Tullis, R. H., Kesimer, M., and Taylor, D. D. (2012) Nanoparticle analysis of circulating cell-derived vesicles in ovarian cancer patients. *Analytical biochemistry* **428**, 44-53

201. Berckmans, R. J., Sturk, A., van Tienen, L. M., Schaap, M. C., and Nieuwland, R. (2011) Cell-derived vesicles exposing coagulant tissue factor in saliva. *Blood* **117**, 3172-3180

202. Fritze, A., Hens, F., Kimpfler, A., Schubert, R., and Peschka-Süss, R. (2006) Remote loading of doxorubicin into liposomes driven by a transmembrane phosphate gradient. *Biochimica et Biophysica Acta (BBA)-Biomembranes* **1758**, 1633-1640

203. Mowrey, D., Chen, Q., Liang, Y., Liang, J., Xu, Y., and Tang, P. (2013) Signal transduction pathways in the pentameric ligand-gated ion channels. *PloS one* **8**, e64326

204. Lacroix, J. J., Campos, F. V., Frezza, L., and Bezanilla, F. (2013) Molecular bases for the asynchronous activation of sodium and potassium channels required for nerve impulse generation. *Neuron* **79**, 651-657

205. Grady, S. R., Drenan, R. M., Breining, S. R., Yohannes, D., Wageman, C. R., Fedorov, N. B., McKinney, S., Whiteaker, P., Bencherif, M., Lester, H. A., and Marks, M. J. (2010) Structural differences determine the relative selectivity of nicotinic compounds for native α4β2*-, α6β2*-, α3β4*-, and α7 nicotinic acetylcholine receptors. *Neuropharmacology* **58**, 1054-1066

206. Walter, N. G., and Bustamante, C. (2014) Introduction to single molecule imaging and mechanics: seeing and touching molecules one at a time. ACS Publications

207. Holzmeister, P., Acuna, G. P., Grohmann, D., and Tinnefeld, P. (2014) Breaking the concentration limit of optical single-molecule detection. *Chemical Society Reviews* **43**, 1014-1028

208. Loveland, A. B., Habuchi, S., Walter, J. C., and van Oijen, A. M. (2012) A general approach to break the concentration barrier in single-molecule imaging. *Nat Meth* **9**, 987-992

209. Chung, H. S., McHale, K., Louis, J. M., and Eaton, W. A. (2012) Single-Molecule Fluorescence Experiments Determine Protein Folding Transition Path Times. *Science* **335**, 981-984

210. Uemura, S., Aitken, C. E., Korlach, J., Flusberg, B. A., Turner, S. W., and Puglisi, J. D. (2010) Real-time tRNA transit on single translating ribosomes at codon resolution. *Nature* **464**, 1012-U1073

211. Pronsato, L., Boland, R., and Milanesi, L. (2013) Non-classical localization of androgen receptor in the C2C12 skeletal muscle cell line. *Archives of Biochemistry and Biophysics* **530**, 13-22

212. Kaznacheyeva, E., Lupu, V. D., and Bezprozvanny, I. (1998) Single-channel properties of inositol (1,4,5)-trisphosphate receptor heterologously expressed in HEK-293 cells. *J Gen Physiol* **111**, 847-856

213. Smith, T. L. (1990) Regulation of intrasynaptosomal free calcium concentrations: studies with the fluorescent indicator, fluo-3. *Neurochemistry International* **16**, 89-94

214. Durisic, N., Godin, A. G., Wever, C. M., Heyes, C. D., Lakadamyali, M., and Dent, J. A. (2012) Stoichiometry of the human glycine receptor revealed by direct subunit counting. *Journal of Neuroscience* **32**, 12915-12920

215. Ulbrich, M. H., and Isacoff, E. Y. (2007) Subunit counting in membrane-bound proteins. *Nat Methods* **4**, 319-321

216. Foo, Y. H., Naredi-Rainer, N., Lamb, D. C., Ahmed, S., and Wohland, T. (2012) Factors affecting the quantification of biomolecular interactions by fluorescence cross-correlation spectroscopy. *Biophys J* **102**, 1174-1183

217. Hastie, P., Ulbrich, M. H., Wang, H. L., Arant, R. J., Lau, A. G., Zhang, Z., Isacoff, E. Y., and Chen, L. (2013) AMPA receptor/TARP stoichiometry visualized by single-molecule subunit counting. *Proceedings of the National Academy of Sciences of the United States of America* **110**, 5163-5168

218. Bharill, S., Fu, Z., Palty, R., and Isacoff, E. Y. (2014) Stoichiometry and specific assembly of Best ion channels. *Proceedings of the National Academy of Sciences of the United States of America* **111**, 6491-6496

219. Haggie, P. M., and Verkman, A. S. (2008) Monomeric CFTR in plasma membranes in live cells revealed by single molecule fluorescence imaging. *The Journal of biological chemistry* **283**, 23510-23513

220. Schillers, H., Shahin, V., Albermann, L., Schafer, C., and Oberleithner, H. (2004) Imaging CFTR: a tail to tail dimer with a central pore. *Cellular physiology and biochemistry : international journal of experimental cellular physiology, biochemistry, and pharmacology* **14**, 1-10

221. Anand, R., Conroy, W. G., Schoepfer, R., Whiting, P., and Lindstrom, J. (1991) Neuronal nicotinic acetylcholine-receptors expressed in *Xenopus* oocytes have a pentameric quaternary structure. *J. Biol. Chem.* **266**, 11192-11198

222. Lukas, R. J., Changeux, J. P., Le Novere, N., Albuquerque, E. X., Balfour, D. J., Berg, D. K., Bertrand, D., Chiappinelli, V. A., Clarke, P. B., Collins, A. C., Dani, J. A., Grady, S. R., Kellar, K. J., Lindstrom, J. M., Marks, M. J., Quik, M., Taylor, P. W., and Wonnacott, S. (1999) International Union of Pharmacology. XX. Current status of the nomenclature for nicotinic acetylcholine receptors and their subunits. *Pharmacol Rev* **51**, 397-401

223. Eaton, J. B., Lucero, L. M., Stratton, H., Chang, Y., Cooper, J. F., Lindstrom, J. M., Lukas, R. J., and Whiteaker, P. (2014) The unique alpha4+/-alpha4 agonist binding site in (alpha4)3(beta2)2 subtype nicotinic acetylcholine receptors permits differential agonist desensitization pharmacology versus the (alpha4)2(beta2)3 subtype. *J Pharmacol Exp Ther* **348**, 46-58

224. Chatterjee, S., Santos, N., Holgate, J., Haass-Koffler, C. L., Hopf, F. W., Kharazia, V., Lester, H., Bonci, A., and Bartlett, S. E. (2013) The alpha5 subunit regulates the expression

and function of alpha4*-containing neuronal nicotinic acetylcholine receptors in the ventral-tegmental area. *PloS one* **8**, e68300

225. Zwart, R., and Vijverberg, H. P. (1998) Four pharmacologically distinct subtypes of a4b2 nicotinic acetylcholine receptor expressed in Xenopus laevis oocytes. *Mol Pharmacol* **54**, 1124-1131.

226. Moroni, M., Zwart, R., Sher, E., Cassels, B. K., and Bermudez, I. (2006) α4β2 nicotinic receptors with high and low acetylcholine sensitivity: pharmacology, stoichiometry, and sensitivity to long-term exposure to nicotine. *Mol Pharmacol* **70**, 755-768

227. Nelson, M. E., Kuryatov, A., Choi, C. H., Zhou, Y., and Lindstrom, J. (2003) Alternate stoichiometries of α4β2 nicotinic acetylcholine receptors. *Molecular Pharmacology* **63**, 332-341

228. Srinivasan, R., Pantoja, R., Moss, F. J., MacKey, E. D. W., Son, C. D., Miwa, J., and Lester, H. A. (2011) Nicotine up-regulates α4β2 nicotinic receptors and ER exit sites via stoichiometry-dependent chaperoning. *Journal of General Physiology* **137**, 59-79

229. Turner, J. R., Castellano, L. M., and Blendy, J. A. (2011) Parallel anxiolytic-like effects and upregulation of neuronal nicotinic acetylcholine receptors following chronic nicotine and varenicline. *Nicotine Tob Res* **13**, 41-46

230. Ngolab, J., Liu, L., Zhao-Shea, R., Gao, G., Gardner, P. D., and Tapper, A. R. (2015) Functional Upregulation of alpha4* Nicotinic Acetylcholine Receptors in VTA GABAergic Neurons Increases Sensitivity to Nicotine Reward. *J Neurosci* **35**, 8570-8578

231. Walsh, H., Govind, A. P., Mastro, R., Hoda, J. C., Bertrand, D., Vallejo, Y., and Green, W. N. (2008) Upregulation of nicotinic receptors by nicotine varies with receptor subtype. *The Journal of biological chemistry* **283**, 6022-6032

232. Mukhin, A. G., Kimes, A. S., Chefer, S. I., Matochik, J. A., Contoreggi, C. S., Horti, A. G., Vaupel, D. B., Pavlova, O., and Stein, E. A. (2008) Greater nicotinic acetylcholine receptor density in smokers than in nonsmokers: a PET study with 2-18F-FA-85380. *Journal of nuclear medicine : official publication, Society of Nuclear Medicine* **49**, 1628-1635

233. Srinivasan, R., Henderson, B. J., Lester, H. A., and Richards, C. I. (2014) Pharmacological chaperoning of nAChRs: a therapeutic target for Parkinson's disease. *Pharmacological Research* **83**, 20-29

234. Fasoli, F., Moretti, M., Zoli, M., Pistillo, F., Crespi, A., Clementi, F., Mc Clure-Begley, T., Marks, M. J., and Gotti, C. (2016) In vivo chronic nicotine exposure differentially and reversibly affects upregulation and stoichiometry of alpha4beta2 nicotinic receptors in cortex and thalamus. *Neuropharmacology* **108**, 324-331

235. Lee, J., and Lee, T.-H. (2017) Single-Molecule Investigations on Histone H2A-H2B Dynamics in the Nucleosome. *Biochemistry* **56**, 977-985

236. Banerjee, P. R., Moosa, M. M., and Deniz, A. A. (2016) Two-Dimensional Crowding Uncovers a Hidden Conformation of alpha-Synuclein. *Angewandte Chemie (International ed. in English)* **55**, 12789-12792

237. Husbands, A. Y., Aggarwal, V., Ha, T., and Timmermans, M. C. (2016) In Planta Single-Molecule Pull-Down Reveals Tetrameric Stoichiometry of HD-ZIPIII:LITTLE ZIPPER Complexes. *The Plant cell* **28**, 1783-1794

238. Rodgers, M. L., Paulson, J., and Hoskins, A. A. (2015) Rapid isolation and single-molecule analysis of ribonucleoproteins from cell lysate by SNAP-SiMPull. *RNA (New York, N.Y.)* **21**, 1031-1041

239. Moonschi, F. H., Effinger, A. K., Zhang, X., Martin, W. E., Fox, A. M., Heidary, D. K., DeRouchey, J. E., and Richards, C. I. (2015) Cell-derived vesicles for single-molecule imaging of membrane proteins. *Angewandte Chemie (International ed. in English)* **54**, 481-484

240. Fox-Loe, A. M., Dwoskin, L. P., and Richards, C. I. (2016) Nicotinic Acetylcholine Receptors as Targets for Tobacco Cessation Therapeutics: Cutting-Edge Methodologies to Understand Receptor Assembly and Trafficking. *Neuromethods* **117**, 119-132

241. Srinivasan, R., Richards, C. I., Xiao, C., Rhee, D., Pantoja, R., Dougherty, D. A., Miwa, J. M., and Lester, H. A. (2012) Pharmacological Chaperoning of Nicotinic Acetylcholine Receptors Reduces the Endoplasmic Reticulum Stress Response. *Molecular Pharmacology* **81**, 759-769

242. Araki, Y., Lin, D. T., and Huganir, R. L. (2010) Plasma membrane insertion of the AMPA receptor GluA2 subunit is regulated by NSF binding and Q/R editing of the ion pore. *Proceedings of the National Academy of Sciences of the United States of America* **107**, 11080-11085

243. Marks, M. J., O'Neill, H. C., Wynalda-Camozzi, K. M., Ortiz, N. C., Simmons, E. E., Short, C. A., Butt, C. M., McIntosh, J. M., and Grady, S. R. (2015) Chronic treatment with varenicline changes expression of four nAChR binding sites in mice. *Neuropharmacology* **99**, 142-155

244. Khiroug, S. S., Pryazhnikov, E., Coleman, S. K., Jeromin, A., Keinanen, K., and Khiroug, L. (2009) Dynamic visualization of membrane-inserted fraction of pHluorin-tagged channels using repetitive acidification technique. *Bmc Neuroscience* **10**

245. Lin, D. T., Makino, Y., Sharma, K., Hayashi, T., Neve, R., Takamiya, K., and Huganir, R. L. (2009) Regulation of AMPA receptor extrasynaptic insertion by 4.1N, phosphorylation and palmitoylation. *Nat. Neurosci.* **12**, 879-U895

246. Zhang, Z., Baksh, M. M., Finn, M. G., Heidary, D. K., and Richards, C. I. (2017) Direct Measurement of Trafficking of the Cystic Fibrosis Transmembrane Conductance Regulator to the Cell Surface and Binding to a Chemical Chaperone. *Biochemistry* **56**, 240-249

247. Foo, Yong H., Naredi-Rainer, N., Lamb, Don C., Ahmed, S., and Wohland, T. (2012) Factors Affecting the Quantification of Biomolecular Interactions by Fluorescence Cross-Correlation Spectroscopy. *Biophysical Journal* **102**, 1174-1183

248. Mazzo, F., Pistillo, F., Grazioso, G., Clementi, F., Borgese, N., Gotti, C., and Francesca Colombo, S. (2013) Nicotine-modulated subunit stoichiometry affects stability and trafficking of α3 β4 nicotinic receptor. *Journal of Neuroscience* **33**, 12316-12328

249. Morales-Perez, C. L., Noviello, C. M., and Hibbs, R. E. (2016) Manipulation of Subunit Stoichiometry in Heteromeric Membrane Proteins. *Structure (London, England : 1993)* **24**, 797-805

250. Henderson, B. J., and Lester, H. A. (2015) Inside-out neuropharmacology of nicotinic drugs. *Neuropharmacology* **96**, 178-193

251. Amici, S. A., McKay, S. B., Wells, G. B., Robson, J. I., Nasir, M., Ponath, G., and Anand, R. (2012) A highly conserved cytoplasmic cysteine residue in the alpha4 nicotinic acetylcholine receptor is palmitoylated and regulates protein expression. *The Journal of biological chemistry* **287**, 23119-23127

252. Barenholz, Y. C. (2012) Doxil®—the first FDA-approved nano-drug: lessons learned. *Journal of controlled release* **160**, 117-134

253. Maruyama, K. (1998) PEG-liposome in DDS and clinical studies. *Nihon rinsho. Japanese journal of clinical medicine* **56**, 632-637

254. Sercombe, L., Veerati, T., Moheimani, F., Wu, S. Y., Sood, A. K., and Hua, S. (2015) Advances and Challenges of Liposome Assisted Drug Delivery. *Frontiers in Pharmacology* **6**

255. Sun, D., Zhuang, X., Xiang, X., Liu, Y., Zhang, S., Liu, C., Barnes, S., Grizzle, W., Miller, D., and Zhang, H.-G. (2010) A novel nanoparticle drug delivery system: the anti-inflammatory activity of curcumin is enhanced when encapsulated in exosomes. *Molecular Therapy* **18**, 1606-1614

256. Ohno, S.-i., Takanashi, M., Sudo, K., Ueda, S., Ishikawa, A., Matsuyama, N., Fujita, K., Mizutani, T., Ohgi, T., and Ochiya, T. (2013) Systemically injected exosomes targeted to EGFR deliver antitumor microRNA to breast cancer cells. *Molecular Therapy* **21**, 185-191

257. Sun, Y., and Liu, J. (2014) Potential of Cancer Cell–Derived Exosomes in Clinical Application: A Review of Recent Research Advances. *Clinical therapeutics* **36**, 863-872

258. Li, W., Li, C., Zhou, T., Liu, X., Liu, X., Li, X., and Chen, D. (2017) Role of exosomal proteins in cancer diagnosis. *Molecular cancer* **16**, 145

259. Tang, M. K., and Wong, A. S. (2015) Exosomes: Emerging biomarkers and targets for ovarian cancer. *Cancer letters* **367**, 26-33

260. Panagiotara, A., Markou, A., Lianidou, E. S., Patrinos, G. P., and Katsila, T. (2017) Exosomes: A Cancer Theranostics Road Map. *Public health genomics* **20**, 116-125

261. Minciacchi, V., Zijlstra, A., Rubin, M. A., and Di Vizio, D. (2017) Extracellular vesicles for liquid biopsy in prostate cancer: where are we and where are we headed? *Prostate cancer and prostatic diseases* **20**, 251

262. Théry, C., Amigorena, S., Raposo, G., and Clayton, A. (2006) Isolation and characterization of exosomes from cell culture supernatants and biological fluids. *Current protocols in cell biology*, 3.22. 21-23.22. 29

263. Fox-Loe, A. M., Moonschi, F. H., and Richards, C. I. (2017) Organelle-specific single-molecule imaging of α4β2 nicotinic receptors reveals the effect of nicotine on receptor assembly and cell-surface trafficking. *Journal of Biological Chemistry* **292**, 21159-21169

264. Fox, A. M., Moonschi, F. H., and Richards, C. I. (2015) The Nicotine Metabolite, Cotinine, Alters the Assembly and Trafficking of a Subset of Nicotinic Acetylcholine Receptors. *Journal of Biological Chemistry* **290**, 24403-24412

265. Moonschi, F. H., Fox-Loe, A. M., Fu, X., and Richards, C. I. (2018) Mammalian Cell-derived Vesicles for the Isolation of Organelle Specific Transmembrane Proteins to Conduct Single Molecule Studies. *Bio-protocol* **8**, e2825

266. Panyam, J., and Labhasetwar, V. (2003) Biodegradable nanoparticles for drug and gene delivery to cells and tissue. *Advanced Drug Delivery Reviews* **55**, 329-347

267. Kohane, D. S. (2007) Microparticles and nanoparticles for drug delivery. *Biotechnology and Bioengineering* **96**, 203-209

268. Nagayasu, A., Uchiyama, K., and Kiwada, H. (1999) The size of liposomes: a factor which affects their targeting efficiency to tumors and therapeutic activity of liposomal antitumor drugs. *Advanced drug delivery reviews* **40**, 75-87

269. Allen, T., and Everest, J. (1983) Effect of liposome size and drug release properties on pharmacokinetics of encapsulated drug in rats. *Journal of Pharmacology and Experimental Therapeutics* **226**, 539-544

270. Heath, T. D., Lopez, N. G., and Papahadjopoulos, D. (1985) The effects of liposome size and surface charge on liposome-mediated delivery of methotrexate-γ-aspartate to cells in vitro. *Biochimica et Biophysica Acta (BBA)-Biomembranes* **820**, 74-84

271. Hobbs, S. K., Monsky, W. L., Yuan, F., Roberts, W. G., Griffith, L., Torchilin, V. P., and Jain, R. K. (1998) Regulation of transport pathways in tumor vessels: role of tumor type and microenvironment. *Proceedings of the National Academy of Sciences* **95**, 4607-4612

272. McKelvey, K. J., Powell, K. L., Ashton, A. W., Morris, J. M., and McCracken, S. A. (2015) Exosomes: mechanisms of uptake. *Journal of circulating biomarkers* **4**, 7

273. Schmidt, T., Schütz, G., Baumgartner, W., Gruber, H., and Schindler, H. (1996) Imaging of single molecule diffusion. *Proceedings of the National Academy of Sciences* **93**, 2926-2929

274. Szebeni, J., and Moghimi, S. M. (2009) Liposome triggering of innate immune responses: a perspective on benefits and adverse reactions: biological recognition and interactions of liposomes. *Journal of liposome research* **19**, 85-90

VITA

Education

08/2012 – present	**University of Kentucky** PhD (expected) in Chemistry
01/2010 – 11/2010	**University of Dhaka** MS in Applied Chemistry and Chemical Engineering
04/2004 – 04/2009	**University of Dhaka** BS in Applied Chemistry and Chemical Engineering

Professional Experience

08/2012 – Present	**Graduate Student,** Department of Chemistry, University of Kentucky, Mentor: Dr. Christopher I. Richards, Assistant Professor of Chemistry
08/2017 – present	**Research Assistant,** Adviser: Dr. Christopher I. Richards, Assistant Professor of Chemistry
07/2016 – 12/2016	**Research Assistant,** Adviser: Dr. Christopher I. Richards, Assistant Professor of Chemistry
08/2014 – 05/2015	**Teaching Assistant,** General Chemistry Laboratory Supervisor: Dr. April French
07/2014 – 08/2014	**Research Assistant,** Adviser: Dr. Christopher I. Richards, Assistant Professor of Chemistry
08/2013 – 05/2014	**Teaching Assistant,** General Chemistry Laboratory Supervisor: Dr. April French
06/2013 – 08/2013	**Research Assistant,** Adviser: Dr. Christopher I. Richards, Assistant Professor of Chemistry
08/2012 – 05/2013	**Teaching Assistant**, General Chemistry Laboratory Supervisor: Dr. April French
05/2012 – 05/2012	**Mathematics Teacher** (Class 5 to 10), Bangladesh International Tutorial, Dhaka, Bangladesh

Awards

06/2016 – 07/2017	Research Challenge Trust Fund Fellowship, Department of Chemistry, University of Kentucky
07/2015 – 06/2016	Graduate School Academic Year fellowship, the Graduate School, University of Kentucky
2016	Max Steckler Fellowship, the Graduate School, University of Kentucky
05/2014 – 06/2014	2014 Center of Membrane Sciences Graduate Student Mentoring Fellowship Merit Award, University of Kentucky
2014	C.H.H. Griffith Outstanding General Chemistry Teaching Assistant Award, Department of Chemistry, University of Kentucky

2010	Master of Science Merit Scholarship, University of Dhaka, Dhaka, Bangladesh
01/2004 – 12/2008	Dutch Bangla Bank Foundation Scholarship, Dutch Bangla Bank Foundation, Dhaka, Bangladesh

Journal Publications

1. **Moonschi, F.H.**, Loe-Fox, A.M., Xu, F., Richards, C.I, "*Mammalian Cell-derived vesicles for the isolation of organelle specific transmembrane proteins to conduct single molecule studies",* Bio-protocol **2018**, 8(9), e2825
2. **Moonschi, F.H.,** Hughes, C.B., Mussmann, G.M., Fowlkes, F., Richards, C.I., Popescu, I., "*Advances in micro- and nanotechnology for the GLP-1-based therapy and imaging of pancreatic beta-cells*", Acta diabetologica, **2018**, 55(5):405-18.
3. Loe-Fox, A.M.*, **Moonschi, F.H.***, Richards, C.I., "*Organelle-specific single-molecule imaging of α4β2 nicotinic receptors reveals the effect of nicotine on receptor assembly and cell-surface trafficking*", J Biol Chem., **2017**, 292(51): 21159-21169.

   ***equal contribution**
4. Fox, A. M., **Moonschi, F.H.**, Richards, C. I. "*The Nicotine Metabolite, Cotinine, Alter s the Assembly and Trafficking of a Subset of Nicotinic Acetylcholine Receptors*", J Biol Chem. **2015**, 290(40):24403-12.
5. **Moonschi, F.H.**, Effinger, A.K., Zhang, X, Martin, W.E., Fox, A.M., Heidary, D.K., DeRouchey, J.E., Richards, C.I., "*Cell-derived vesicles for single-molecule imaging of membrane proteins*". Angew Chem Int Ed Engl. **2015**, 54(2):481-4.