MEASUREMENT AND ANALYSIS OF CALCIUM-DEPENDENT EXOCYTOSIS IN GIANT EXCISED MEMBRANE PATCHES

APPROVED BY SUPERVISORY COMMITTEE

_____

Ege T. Kavalali, Ph.D., Committee Chair

_____

Donald W. Hilgemann, Ph.D., Advisor

_____

Thomas C. Südhof, M.D.

_____

Josep Rizo, Ph.D.

To my parents, Hsien-Yi and Mei-Hsia

MEASUREMENT AND ANALYSIS OF CALCIUM-DEPENDENT EXOCYTOSIS IN GIANT EXCISED MEMBRANE PATCHES

by

TZU-MING WANG

DISSERTATION

Presented to the Faculty of the Graduate School of Biomedical Sciences

The University of Texas Southwestern Medical Center at Dallas

In Partial Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

The University of Texas Southwestern Medical Center at Dallas

Dallas, Texas

May, 2008

Acknowledgements

First, I would like to give my thanks to my graduate mentor, Dr. Donald W. Hilgemann, for giving me 100% freedom to do what I want to do, and insightful suggestions and unlimited support during these years. He is more like a good friend of mine rather than a big boss on top of me. I would also like to thank Dr. Thomas C. Südhof for giving me the opportunity to learn with him. In his lab, I learned not only the techniques, but also the cautious attitude a good scientist shall have. I must also give my thanks to my thesis committee members, Drs. Ege. T. Kavalali and Josep Rizo, for their criticism and advice, which guided me toward the completion of my dissertation.

I would like to thank my friends in Don and Tom's labs, especially Chengcheng Shen, Marc Llaguno, Alp Yaradanakul, and Vincenzo Lariccia. Chengcheng helped me a lot in computer programming, Marc gave me many critical suggestions for making carbon electrode, and Alp and Vincenzo are the persons who make the lab like a “family”.

I would also like to thank my sister and her husband for taking care of my parents over these years. Most of all, I would like to give my thanks to my parents and my wife. Without their love and encouragement they have given to me, it would be impossible for me to complete my dissertation. They give me the courage and strength to face any coming challenges in my life.

MEASUREMENT AND ANALYSIS OF CALCIUM-DEPENDENT EXOCYTOSIS IN GIANT EXCISED MEMBRANE PATCHES

Publication No. _____

Tzu-Ming Wang Ph.D.

The University of Texas Southwestern Medical Center at Dallas, 2008

Advisor: Donald W. Hilgemann Ph.D.

Ca^{2+}-dependent exocytosis was studied in both excised and whole-cell patch clamp with emphasis on the rat secretory cell line, RBL. Capacitance and amperometric recordings show that secretory granules (SGs) containing serotonin are mostly lost from excised patches. Small vesicles that are retained (non-SGs) do not contain substances detected by amperometry. Non-SG fusion is reduced by tetanus toxin light chain treatment, however, it is unaffected by *N*-ethylmaleimide, implying that SNARE cycling

is not required for non-SG fusion in excised patches. Although non-SG fusion is ATP-dependent and blocked by PI-kinase inhibitors, wortmannin and adenosine, the dependency is not neutralized by the PI(3)-kinase inhibitor LY294002, $PI(4,5)P_2$ ligands, such as neomycin, a PI-transfer protein that can remove PI from membranes, and $PI(4,5)P_2$, PI(3)P and PI(4)P antibodies etc. In whole-cell recording, non-SG fusion is strongly reduced by osmotically-induced cell swelling, and subsequent recovery after shrinkage is inhibited by wortmannin, indicating that membrane stretch occurring during patch formation may be a major cause of the ATP-dependency in excised patches.

Syt7 and several PLCs are not required for non-SG fusion because fusion remains robust in mouse embryonic fibroblasts deficient of Syt7, PLCδ1, PLCδ1/δ4, or PLCγ1. Furthermore, the Ca^{2+} dependence of non-SG fusion reflects a lower Ca^{2+} affinity (K_D ~71 μM) than expected for these C2-domain-containing proteins.

I also developed a program for measuring and analyzing membrane capacitance. The program uses either sine waves or square waves to estimate cell parameters. Phase-sensitive detection is utilized in both cases. For square wave perturbation, either integrated charges or direct current trace is used for calculating cell parameters. Other functions like digital filtering, pulse stimulation, offline phase angle adjustment, baseline subtraction, and data normalization are also implemented.

In summary, using the software I developed, non-SG fusion were characterized and found to be regulated substantially differently from SG fusion. An ATP-dependent

process is probably required for restoring non-SG fusion capability after it is perturbed by membrane stretch and dilation.

Table of Contents

Committee signatures....i
Dedication....ii
Title page....iii
Copyright....iv
Acknowledgements....v
Abstract....vi
Table of contents....ix
Prior publications....xiv
List of figures....xv
List of tables....xviii
List of abbreviations and symbols....xix

Chapter 1. General introduction

1.1 Proteins involved in membrane fusion....1
SNAREs and SNARE cycle....1
Munc18....2
Synaptotagmin 1....3

1.2 Lipids and lipases implicated in fusion....5
$PI(4,5)P_2$ and other lipids....5
Phospholipase Cs....6
Phospholipase D....8
Phospholipase A_2....9

Chapter 2. Fusion of artificial liposomes to excised patches

2.1 Introduction....18

2.2 Methods....22
Small-scale plasmid DNA preparation....22
SDS-PAGE analysis....23
Western blot....25

Dot blot – *Express*....26
Purification of Synaptotagmin 1 from Sf9 cells....26
Purification of Synaptobrevin 2 from *E. coli*....29
Reconstitution of Syt1 and Syb2 into liposomes....30
Quality assays of the reconstituted proteoliposomes....32
Triple-marker expression system for Stx1A, SNAP25A, and Munc18-1....33
Liposome perfusion and capacitance measurement in excised patches....35
2.3 Results and discussion....37
Purification of Synaptotagmin 1 37
Purification of Synaptobrevin 2....38
Quality of the liposomes....39
Establishment of the triple-marker transient expression system....41
Perfusion of artificial liposomes to excised patches....43
Chapter 3. Fusion of endogenous vesicles in excised patches
3.1 Abstract....59
3.2 Introduction....61
3.3 Methods....65
Cell culture....65
Solutions....65
Recording software....67
Patch clamp and data acquisition....70
Patch amperometry....71
Recombinant proteins and antibodies....72
Data analysis....73
3.4 Results....75
Distinct vesicle populations in RBL cells....75
Non-SG fusion is SNARE-dependent but NEM-insensitive in excised patches....76
ATP hydrolyzing processes support non-SG fusion....77
ATP-dependent generation of $PI(4,5)P_2$ is not critical....78

Non-SG fusion is probably phosphatydialinositide-independent.....79
Wortmannin/adenosine-insensitivity of non-SG fusion in whole-cell recording.....80
Non-SG fusion in whole-cell recording is blocked by cell swelling.....81
Synaptotagmin VII and PLCδs are not required for non-SG fusion.....83
Ca^{2+}-dependence of non-SG fusion in excised RBL patches.....84
3.5 Discussion.....86
Membrane fusion in excised giant membrane patches.....86
Non-secretory fusion in excised patches.....87
ATP-sensitivity of non-SG fusion.....88
Wound repair and non-SG fusion.....89
3.6 Some more methodological details.....92
Isolation of rodent peritoneal mast cells.....92
Tips for reducing noise in patch amperometric recordings.....93
Tips for making carbon electrodes.....97
Efforts for preserving secretory vesicles in excised patches.....101
Chapter 4. Software and algorithm development
4.1 Introduction.....118
4.2 Algorithms for capacitance measurement.....123
Phase-sensitive detection.....123
Phase-sensitive detection – online calculation of the phase angle.....125
Phase-sensitive detection – offline adjustment of the phase angle.....126
Phase-sensitive detection – computer implementation.....127
Square-wave perturbation – based on fitting the current transient.....128
I-SQA – computer implementation.....134
Square-wave perturbation – based on fitting the transferred charges.....139
Q-SQA – computer implementation.....144
Square-wave perturbation – validating the algorithms using Simulink.....146
Square-wave perturbation – PSD analysis of SQA data.....148

4.3 Capmeter 6.....**151**
What is it?.....151
System requirements.....151
Connection diagram.....152
Things you need to know before using.....153
Running the program.....154
Using PSD.....157
Using SQA.....159
Using digital filters.....160
Variables in the Workspace.....160
Subtracting the baseline.....163
Scaling the data.....164
Showing the data.....165
Exporting the data.....166
Giving pulses.....167
TTL triggering.....168
Reader mode.....169
The information flow.....169
Main variables in the program.....171
4.4 Capmeter 1.....**174**
What is it?.....174
Connection diagram.....174
Things you need to know before using.....174
4.5 IQplot.....**175**
What is it?.....175
Connection diagram.....175
Things you need to know before using.....175
Running the program.....177
Giving pulses and applying notes.....178

Methods for charge integration....180
Display modes – online....181
Display modes – offline....182
Variables in the Workspace....184
The information flow....186
Main variables in the program....188
4.6 Dynamically linked subroutines....190
CapEngine4.mexw32....190
Dfilter.mexw32....191
Dfilter2.mexw32....192
DispCtrl.mexw32....193
IQlizer.mexw32....194
PhaseMatcher2.mexw32....195
SqWaveCalc.mexw32....196
4.7 Capmodule4....197
What is it?....197
Things you need to know before using....197
Running the program....197
Hot keys....198
The information flow....199
The sound sequence....200
4.8 SlopeScan....202
What is it?....202
How to use it?....202
Bibliography....215

Prior Publications

Wang TM and Hilgemann DW. 2008. Ca-dependent non-secretory vesicle fusion in a secretory cell. *J Gen Physiol*. In press.

Yaradanakul A, Wang TM, Lariccia V, Lin MJ, Shen C, Liu X, and Hilgemann DW. 2008. Massive Ca-induced membrane fusion and phospholipid changes triggered by reverse Na/Ca exchange in BHK fibroblasts. *J Gen Physiol*. In press.

Chen X, Arac D, Wang TM, Gilpin CJ, Zimmerberg J, and Rizo J. 2006. SNARE-mediated lipid mixing depends on the physical state of the vesicles. *Biophys J.* 90:2062-2074.

Wang TM, Chen YH, Liu CF, Tsai HJ. 2002. Functional analysis of the proximal promoter regions of fish rhodopsin and myf-5 genes using transgenesis. *Mar Biotechnol* (NY). 4:247-255.

Ma GC, Wang TM, Su CY, Wang YL, Chen S, and Tsai HJ. 2001. Retina-specific cis-elements and binding nuclear proteins of carp rhodopsin gene. *FEBS Lett.* 508:265-271.

List of Figures

Figure 1.1 SNAREs and other proteins involved in exocytic and endocytic pathways.....11

Figure 1.2 The neuronal SNARE complex.....12

Figure 1.3 The SNARE cycle in synaptic vesicle exocytosis.....13

Figure 1.4 Different models of SM protein-syntaxin interaction.....14

Figure 1.5 PI distribution of endo- and exocytic pathways.....15

Figure 1.6 Functions of $PI(4,5)P_2$.....16

Figure 1.7 Cleavage sites of four classes of phospholipases.....17

Figure 2.1 Chromatographic purification of untagged Syt1.....45

Figure 2.2 Purification of $(His)_6$-Syt1.....46

Figure 2.3 Purification of HBM-$(His)_8$-Syt1 and Syt1-$(His)_8$.....47

Figure 2.4 Estimation of HBM-$(His)_8$-Syt1 and Syt1-$(His)_8$ concentration.....48

Figure 2.5 Purification of Syb2.....49

Figure 2.6 Reconstitution of Syt1 at different OG concentrations.....50

Figure 2.7 Size distribution of reconstituted liposomes.....51

Figure 2.8 Leakage assay of Syt1-containing liposomes.....52

Figure 2.9 Orientation tests of Syt1/Syb2 liposomes.....53

Figure 2.10 The triple-marker transient expression system.....54

Figure 2.11 Stable cell lines co-expressing Stx1A, SNAP25A, and Munc18-1.....55

Figure 2.12 Perfusion of artificial liposomes to excised patches.....56

Figure 3.1. Method to determine whole-cell capacitance via square wave perturbation.....

.....107

Figure 3.2. Amperometric and capacitance measurement in RBL cells.....108

Figure 3.3. Non-SG fusion is SNARE-dependent.....109

Figure 3.4. ATP hydrolysis is required for supporting non-SG fusion.....110

Figure 3.5. Non-SG fusion in excised patches is wortmannin/adenosine-sensitive.....111

Figure 3.6. Non-SG fusion is not blocked by antibodies against PI(3)P and PI(4)P.....112

Figure 3.7. Non-SG fusion in whole-cell recording is wortmannin/adenosine insensitive…

.....113

Figure 3.8. Non-SG fusion in whole-cell recordings is strongly inhibited by cell swelling..

.....114

Figure 3.9. Synaptotagmin VII is not required for non-SG fusion.....115

Figure 3.10. Ca-dependence of non-SG fusion in excised patches from RBL cells.....116

Figure 4.1. Geometrical view of phase-sensitive detection.....204

Figure 4.2. Online calculation of the phase angle.....205

Figure 4.3. Offline adjustment of the phase angle.....206

Figure 4.4. Demonstration of phase-sensitive detection.....207

Figure 4.5. Square-wave perturbation – based on fitting the current transient.....208

Figure 4.6. Square-wave perturbation – based on fitting the transferred charges.....209

Figure 4.7. Correction of the integrated charges.....210

Figure 4.8. Demonstration of the charge correction.....211

Figure 4.9. Generation of the model current using Simulink.....212

Figure 4.10. Some more about the diagram in Simulink.....213

Figure 4.11. Square-wave perturbation – PSD analysis of SQA data.....214

List of Tables

Table 3.1. Effects of various reagents on non-SG fusion in excised patches.....117

List of Abbreviations and Symbols

AA	arachidonic acid
AI	analogue input
alpha, α	phase shift
AMP-PNP	adenosine 5′-(β,γ-imido)triphosphate
AO	analogue output
ATP	adenosine 5′-triphosphate
BHK	baby hamster kidney
BSA	bovine serum albumin
C	capacitance, coulomb
Ca	calcium
CAPS	calcium-dependent activator protein for secretion
Ch	channel
CHO	Chinese hamster ovary
Cm	membrane capacitance
CM	carboxymethyl
CPU	central processing unit (of the computer)
DAG	diacylglycerol
DAQ	data acquisition
DEAE	diethylaminoethyl
delta, Δ	differences
DNA	deoxyribonucleic acid
DOPS	1,2-dioleoyl-*sn*-glycero-3-phosphoserine
DSP	digital signal processing
EDTA	ethylene diamine tetraacetic acid
EGTA	ethylene glycol-bis (β-aminoethyl ether)-tetraacetic acid
ER	endoplasmic reticulum

f	frequency
FBS	fetal bovine serum
FRET	fluorescence resonance energy transfer
G	conductance
GST	glutathione S-transferase
GTP	guanosine 5'-triphosphate
GUI	graphic-user interface
HA	hydroxyapatite
HEPES	4-(2-Hydroxyethyl)piperazine-1-ethanesulfonic acid
I	current
IP_3	inositol triphosphate
IPTG	isopropyl β-D-1-thiogalactopyranoside
IRES	internal ribosomal entry site
K_D	dissociation-constant
LB	Luria-Bertani broth
LMV	large multilamellar vesicle
LPL	lysophospholipid
LUV	large unilamellar vesicle
MAPK	mitogen-activated protein kinas
MEF	mouse embryo fibroblast
MES	2-(*N*-morpholino)ethanesulfonic acid
MLCK	myosin light chain kinase
Munc	mammalian homologue of Unc
NaN	not a number
NEM	*N*-ethylmaleimide
Ni	nickle
NLS	nuclear localization signal
NMG	*N*-methyl-D-glucamine
NSF	*N*-ethylmaleimide-sensitive factor
NTA	nitrilotriacetic acid

OD	optical density
OG	octyl β-D-glucopyranoside
PA	phosphatidic acid
PAGE	polyacrylamide gel electrophoresis
PBS	phosphate-buffered saline
PC	phosphatidylcholine
PCR	polymerase chain reaction
PH	pleckstrin homology
PI	phosphatidylinositol
PI(3)K	PI(3) kinase
PI(3)P	phosphatidylinositol 3-phosphate
$PI(4,5)P_2$	phosphatidylinositol 4,5-bisphosphate
PIP_3	phosphatidylinositol 3,4,5-triphosphate
PKC	protein kinase C
PLA_2	phospholipase A_2
PLC	phospholipase C
PLD	phospholipase D
PMA	phorbol 12-myristate 13-acetate
PMSF	phenylmethylsulphonyl fluoride
POPC	1-palmitoyl-2-oleoyl-*sn*-glycero-3-phosphocholine
PSD	phase-sensitive detection or detector
PTK	protein tyrosine kinase
PTS1	peroxisomal targeting sequence 1
Q	charges
Qs	charges, steady-state-current-subtracted
Ra	access resistance
RAM	random access memory
RBL	rat basophil leukemia
Ref	reference
RIM	Rab3-interacting molecule

Rm	membrane resistance
SDS	sodium dodecylsulfate
SG	secretory granule
SQA	square-wave algorithm
SM	Sec1/Munc18-like
SNAPs	soluble NSF attachment proteins
SNAP25	Synaptosome-associated protein of 25 kDa
SNARE	SNAP receptor
Stx	Syntaxin
Syb	Synaptobrevin
Syt	Synaptotagmin
omega, ω	angular speed
t	time
tau, τ	time constant
TBST	Tris-buffered saline with Tween 20
TEMED	*N*,*N*,*N*,*N*-tetramethyl-ethylenediamin
TeTx	tetanus toxin
theta, θ	angle
TIR-FM	total internal reflection fluorescence microscopy
Tris	tris(hydroxymethyl)aminomethane
t-SNARE	target membrane SNARE
V	voltage, signal amplitude
Vc	command voltage
Vss	steady-state potential
v-SNARE	vesicle SNARE
wort	wortmannin

Chapter 1. General introduction

1.1 Proteins involved in membrane fusion

SNAREs and SNARE cycle

Membrane fusion is a fundamental biophysical phenomenon that is utilized and tightly regulated in a living organism. Neurotransmitters are released upon vesicle fusion, and by means of the tightly controlled fusion machinery, the release of neurotransmitters is spatially restricted to the active zone of the presynaptic terminal, and temporally controlled upon stimuli.

Fusion between the vesicle and plasma membranes requires the interaction between these two lipid bilayers. In general, it is wildly accepted that the process (at least in part) is mediated by the SNARE (soluble *N*-ethylmaleimide-sensitive factor attachment protein receptor) proteins, which are evolutionarily conserved among species and are utilized in both exocytic and endocytic pathways (Fig. 1.1) (Jahn et al., 2003). Synaptobrevin (Syb) is a SNARE protein that is expressed on the vesicle (v-SNARE), and syntaxin (Stx) and SNAP25 (synaptosomal-associated protein of 25 kDa) are distributed on the target membrane (t-SNAREs). These three components can form a four-helix bundle (Fig. 1.2), termed the SNARE complex or core complex, and bring the vesicle and membranes to close proximity (Rizo, 2003; Rizo and Südhof, 2002).

Upon membrane fusion, the trans-SNARE complex (three of its components are in two opposing bilayers) becomes cis-SNARE complex (all three components are in the same bilayer), and the complex is then unwound in an ATP-dependent process mediated

by NSF (*N*-ethylmaleimide-sensitive factor) and SNAPs (soluble NSF attachment proteins) (Rizo and Südhof, 2002). The recycled SNAREs are now ready for the next round of fusion (Fig. 1.3).

Munc18

In addition to SNAREs, Munc18, a mammalian homologue of *C. elegans* UNC-18, is also essential for synaptic vesicle fusion. In Munc18 knockout mice, the Ca^{2+}-triggered release, minis, and α-latrotoxin-induced exocytosis were abolished (Verhage et al., 2000). However, it was reported that Munc18 binds to the "closed" Stx (Fig. 1.4, upper panel), which is incompatible with the core complex, and competes with the formation of the complex (Dulubova et al., 1999).

In spite of these controversial results, two other active zone proteins, Munc 13 (mammalian homologue of *C. elegans* UNC-13) and RIM (Rab3-interacting molecule) were proposed to release Munc18 from Stx and facilitate the formation of the core complex. In UNC-13 mutant, neurotransmitter release was completely blocked, and RIM mutant in worms showed severe decrease in neurotransmitter release. Furthermore, the UNC-13 and RIM mutants can be rescued by expressing the open form of Stx (Koushika et al., 2001; Richmond et al., 2001).

It is attracting that Stx is blocked by Munc18 and released in the active zone, however, it cannot explain some controversial experimental observations. As mentioned previously, neurotransmitter release was abolished rather than increased in Munc18 knockout mice (Verhage et al., 2000). In addition, Sec1, a Munc18 homologue in yeast, binds to Stx in

assembled SNARE complex rather than the closed Stx (Carr et al., 1999). Nevertheless, in other vesicular transport systems such as ER and Golgi, Stx homologues do not form the closed conformation and the Sec1/Munc18 homologues bind to the N-terminus of Stx (Fig. 1.4, middle panel) (Yamaguchi et al., 2002). Recently, it was reported that Munc18 does bind to the assembled SNARE complex (Fig. 1.4, lower panel) (Dulubova et al., 2007), and the two different interacting modes (open and closed Stx) are essential for synaptic vesicle fusion, and are coupled by functionally critical binding to Stx N-terminus (Khvotchev et al., 2007).

Synaptotagmin 1

Synaptotagmin (Syt) 1 is a type I transmembrane protein with its N-terminus in the lumen of the synaptic vesicle. It contains a N-terminal transmembrane region, and two C2 domains in its C-terminus, and was proposed to be a potential Ca^{2+} sensor in regulated exocytosis (Perin et al., 1990). Further experiments indicated that Syt1 binds to negatively charged phospholipids in a Ca^{2+}-dependent manner at physiological Ca^{2+} concentrations (Brose et al., 1992), and the Ca^{2+}-binding affinities are well correlated with the Ca^{2+} sensitivities of neurotransmitter release (Fernández-Chacón et al., 2002; Rhee et al., 2005). In Syt1 knockout mice, the fast synchronous neurotransmitter release was selectively attenuated (Geppert et al., 1994), indicating that Syt1 is crucial for this type of membrane fusion. Together with the evidence mentioned above, it is now widely accepted that Syt1 is the Ca^{2+} sensor for fast synchronous neurotransmitter release in neurons.

It is still not totally clear how Syt1 triggers membrane fusion in a Ca^{2+}-dependent

manner. As mentioned previously, the Ca^{2+}-binding affinities and the Ca^{2+} sensitivities of neurotransmitter release are well correlated (Fernández-Chacón et al., 2001; Rhee et al., 2005). Apparently, the Ca^{2+}-dependent phospholipid binding must play certain roles in triggering fusion, and it is proposed recently that Syt1, together with SNAREs, may bring two opposing lipid bilayers very close to each other and accelerate membrane fusion (Rizo et al., 2006). In addition to Ca^{2+}-dependent phospholipid binding, Syt1 also interacts with SNAREs in both Ca^{2+}-dependent and -independent manners (Südhof and Rizo, 1996). As suggested recently, the Ca^{2+}-induced displacement of complexin (a protein that binds SNARE complex) from SNARE complex might account for the triggering of fast neurotransmitter release in neurons (Tang et al., 2006).

1.2 Lipids and lipases implicated in fusion

$PI(4,5)P_2$ and other lipids

Phosphatidylinositides (PIs) and their derivatives are presently thought to be critical regulators of membrane trafficking in the cells (Fig. 1.5) (De Matteis and Godi, 2004). Phosphatidylinositol 4,5-bisphosphate ($PI(4,5)P_2$) is one of the most versatile PIs that has been implicated in a variety of different physiological activities, including membrane fusion (Fig. 1.6) (McLaughlin and Murray, 2005). Evidence from PC12 cells suggested that formation of $PI(4,5)P_2$ microdomains at syntaxin clusters can activate the exocytotic sites (Aoyagi et al., 2005). In addition, $PI(4,5)P_2$ appears to be required for "priming" of vesicles in pancreatic β-cells (Olsen et al., 2003) and the yeast homotypic vacuole fusion system (Mayer et al., 2000). In dense core vesicle fusion, $PI(4,5)P_2$ promotes the binding of CAPS (calcium-dependent activator protein for secretion) to the plasma membrane and increases the initial fusion rate (Grishanin et al., 2004; Loyet et al., 1998), and it is also reported that $PI(4,5)P_2$ regulates the releasable vesicle pool size in chromaffin cells (Milosevic et al., 2005). Moreover, $PI(4,5)P_2$ is suggested to promote fusion directly via its interaction with Syt1 in a liposome-liposome fusion system (Bai et al., 2004), and it also modulates many enzymes that are implicated in regulating fusion (see next section), including phospholipase D (PLD), phospholipase A_2 (PLA_2) etc. (Rhee, 2001). Finally, metabolites of phospholipase Cs (PLCs) cleavage of $PI(4,5)P_2$, i.e. diacylglycerol (DAG) and inositol triphosphate (IP_3), are also implicated in regulation of some vesicle fusion processes (Jun et al., 2004).

Other PIs, such as phosphatidylinositol 3,4,5-triphosphate (PIP_3) and

phosphatidylinositol 3-phosphate (PI(3)P) are also implicated in regulating membrane trafficking, including events leading up to fusion (Lindmo and Stenmark, 2006). For example, inhibition of a class IA PI(3) kinase (PI(3)K), which mainly produces PIP_3, reduces receptor-mediated degranulation in mast cells (Ali et al., 2004). In addition, a synapsin I-associated PI(3)K is implicated in mediating delivery of synaptic vesicles to the readily releasable pool in neurons (Cousin et al., 2003). Furthermore, PI(3)K-C2α, which produces mainly PI(3)P, is also required for the ATP-dependent priming of secretory granules (SGs) in neurosecretory cells (Meunier et al., 2005), and when PI(3)P was sequestered by PI(3)P binding motifs (tandem repeats of FYVE motif), secretion was also abolished (Meunier et al., 2005).

Some other lipids and lipid derivatives are also implicated in regulating the fusion process. Cholesterol is reported to be critical for the clustering of SNAREs in the fusion hot spots (Lang et al., 2001). DAG, a metabolite of $PI(4,5)P_2$, can activate Munc13 (mammalian homologue of UNC-13) and potentiate membrane fusion (Rhee et al., 2002). Finally, arachidonic acid (AA), which is also a $PI(4,5)P_2$ metabolite, is reported to facilitate SNARE complex formation in the presence of Munc18 (Rickman and Davletov, 2005), and potentiate exocytosis in chromaffin cells (Latham et al., 2007).

Phospholipase Cs

Phospholipase C is an enzyme family that contains X and Y catalytic domains, and depending on the subtypes, it also contains other domains like C2, EF-hand, SH, and PH domains etc. (Rhee, 2001). The classical substrate of PLCs is $PI(4,5)P_2$, and the resulting products, DAG and IP_3 (Fig. 1.7), are utilized to regulate a variety of different cellular

activities. While IP_3 increases intracellular Ca^{2+} concentration via Ca^{2+} release from internal stores, DAG also plays several critical roles in regulating membrane fusion. In yeast homotypic vacuole fusion system, sequestering DAG by applying DAG-binding C1B domain blocked vacuole fusion (Jun et al., 2004; Thorngren et al., 2004), and similar effects were observed when the PLC inhibitors 3-nitrocoumarin and U73122 were added into the reaction mixture (Jun et al., 2004). In addition to the well-known DAG target, protein kinase C (PKC), DAG also interacts with Munc13 and augments neurotransmitter release (Rhee et al., 2002), and its analogue, PMA (phorbol 12-myristate 13-acetate), has also been shown to increase the primed SGs in chromaffin cells (Gil et al., 2001). Furthermore, without interacting with other protein factors, DAG itself is enough to promote membrane fusion between synthetic lipid bilayers (Goñi and Alonso, 1999). All of the evidence suggests that the activities of PLC and the production of DAG are deeply involved in regulating and/or mediating membrane fusion.

Among the PLC family, PLCδ is the one that is most sensitive to Ca^{2+} activation (Rhee, 2001). An intracellular Ca^{2+} concentration of 0.1-10 μM is enough to stimulate the activity of PLCδ1 (Allen et al., 1997), and it is also localized to the plasma membrane (Lee et al., 2004). Thus, the hypothesis that PLCδ1 triggered certain forms of membrane fusion upon Ca^{2+}-dependent DAG production becomes very attractive. While the PLCδ1 knockout mice showed problems of skin stem cell lineage commitment (Nakamura et al., 2003), knockout of another PLCδ isoform, PLCδ4, did show deficient acrosome reaction, which is an exocytotic event in sperm required for fertilization (Fukami et al., 2001).

Another PLC that comes into the scope is PLCγ. PLCγ can be activated through several ways. It can be activated by receptor protein tyrosine kinases (PTKs), nonreceptor PTKs, and lipid-derived messengers like phosphatidic acid (PA) and AA (Rhee, 2001). Both PA and AA are implicated in regulating membrane fusion (Blackwood et al., 1997; Latham et al., 2007; Rickman and Davletov, 2005). One interesting feature of PLCγ is that it hydrolyzes mainly PI, but not $PI(4,5)P_2$ (Mitchell et al., 2001), and the activity is blocked by the PI(3)K inhibitors wortmannin and LY294002 (Mitchell et al., 2001).

Phospholipase D

PLD is a class of phospholipases that cleaves phosphatidylcholine (PC) and produces PA and choline (Fig. 1.7). PA itself might be fusogenic because it is cone-shaped. Dephosphorylation of PA by PA phosphohydrolase produces DAG (side chains are saturated/mono-unsaturated), which is fusogenic, too. PLD is regulated by $PI(4,5)P_2$ (Pertile et al., 1995), Ca^{2+} (Qin et al., 1997), and small G proteins (Caumont et al., 1998), and its activity is also implicated in regulating exocytosis (Choi et al., 2002). In chromaffin cells, activation of PLD near the exocytotic sites is found to be important for catecholamine release (Caumont et al., 2000), and a similar finding has also been reported in PC12 cells (Vitale et al., 2005). Moreover, blocking PLD activities by 1-butanol or by inactive mutant PLD suppressed secretion in mast cells (Choi et al., 2002), and the PLD isoforms PLD1 and PLD2 were found to regulate different phases of exocytosis (Choi et al., 2002).

Phospholipase A_2

PLA_2 is a phospholipase superfamily that cleaves phospholipid and produces lysophospholipid (LPL) and free fatty acid (Fig. 1.7). It can be found in both secretory or cytoplasmic forms (Brown et al., 2003), and their activities may also require Ca^{2+} depending on the subgroups (Dennis, 1994). The activity of PLA_2 has been reported to facilitate membrane fusion. In a hemi-reconstituted cell-free system, the Ca^{2+}-dependent activity of PLA_2 is reported to promote chromaffin SG fusion in a Ca^{2+}-independent manner (Karli et al., 1990). A similar effect has also been observed by pretreating plasma membrane with PLA_2, and the fusion step was also Ca^{2+}-independent (Nagao et al., 1995). Furthermore, the activities of phospholipases, including PLA_2, are reported to stimulate degranulation of permeabilized RBL mast cells (Cohen and Brown, 2001).

If $PI(4,5)P_2$ is the substrate for PLA_2, the resulting free fatty acid is AA, which is a precursor of some inflammatory mediators (Brown et al., 2003; Dennis, 1994). As mentioned earlier, AA has also been implicated in potentiating membrane fusion (Latham et al., 2007), and it facilitates SNARE complex formation in the presence of Munc18 (Latham et al., 2007; Rickman and Davletov, 2005) and promotes interaction between SNARE complex and Munc18 through Stx (Latham et al., 2007). The other PLA_2 cleavage product is LPL, which was thought to be fusogenic (Poole et al., 1970), and introducing LPL to membrane vesicles prepared from rat parotid acinar cells increased fusion between membrane vesicles and SGs (Nagao et al., 1995). However, direct involvement of LPL in triggering fusion has been ruled out in a hemi-reconstituted cell-free system (Karli et al., 1990), and LPL has also been shown to inhibit fusion of

biological membranes (Chernomordik et al., 1993) and synthetic liposomes (Chen et al., 2006). In general, it is now believed that LPL inhibits membrane fusion by preventing the formation of fusion intermediates such as stalks (Chernomordik et al., 1993). If LPL plays any regulatory role in membrane fusion in vivo is still not clear.

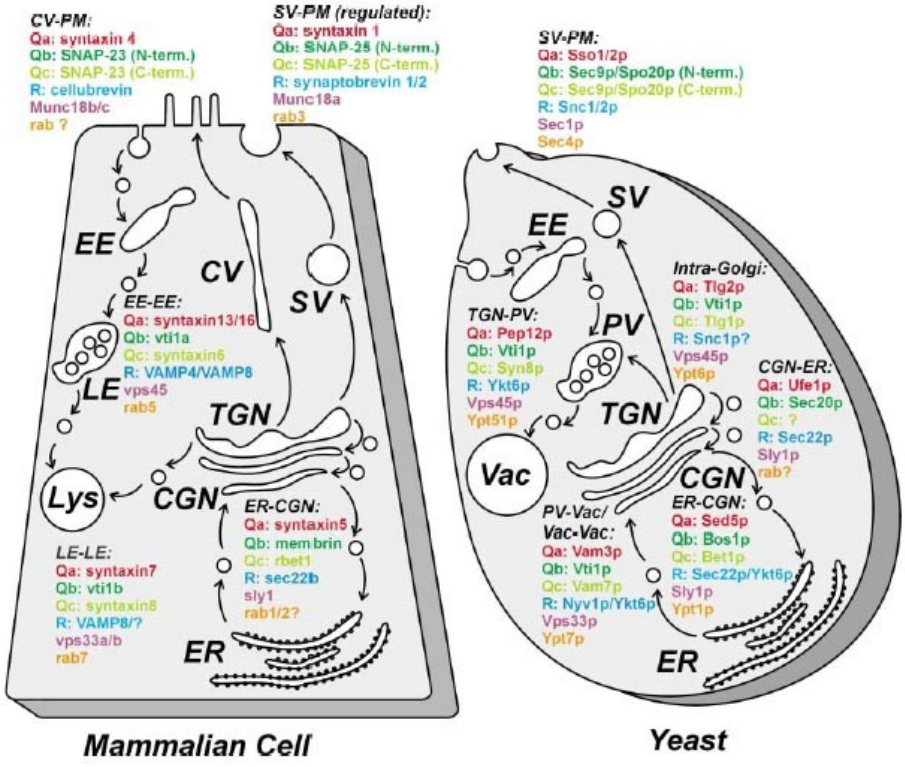


Figure 1.1 SNAREs and other proteins involved in exocytic and endocytic pathways. SNAREs forming the four-helix bundle (Qa, Qb, Qc, R), SM (Sec1/Munc18-like) proteins, and rab proteins are shown. EE: early endosome; CV: constitutive vesicle; SV: secretory vesicle; LE: late endosome; Lys: lysosome; TGN: trans-Golgi network; CGN: cis-Golgi network; ER: endoplasmic reticulum; PV: prevacuolar compartment (corresponding to late endosome); Vac: vacuole (corresponding to lysosome). (Fig. 6 of Jahn et al., 2003).

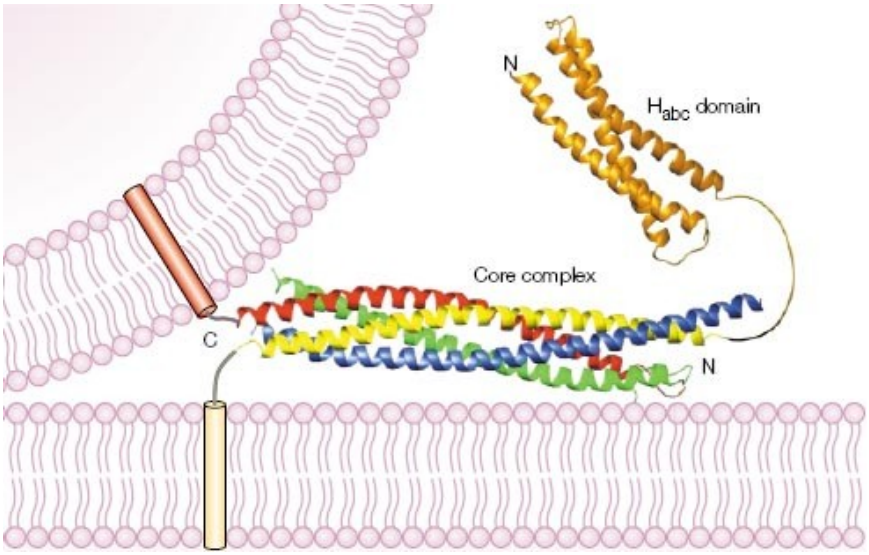


Figure 1.2 The neuronal SNARE complex. The crystal structure of the SNARE complex and the NMR structure of syntaxin 1 Habc domain are shown. Red: synaptobrevin; yellow: syntaxin 1; blue: SNAP25 N-terminus; green: SNAP25 C-terminus; orange: Habc domain of syntaxin 1. (Fig. 2 of Rizo and Südhof, 2002).

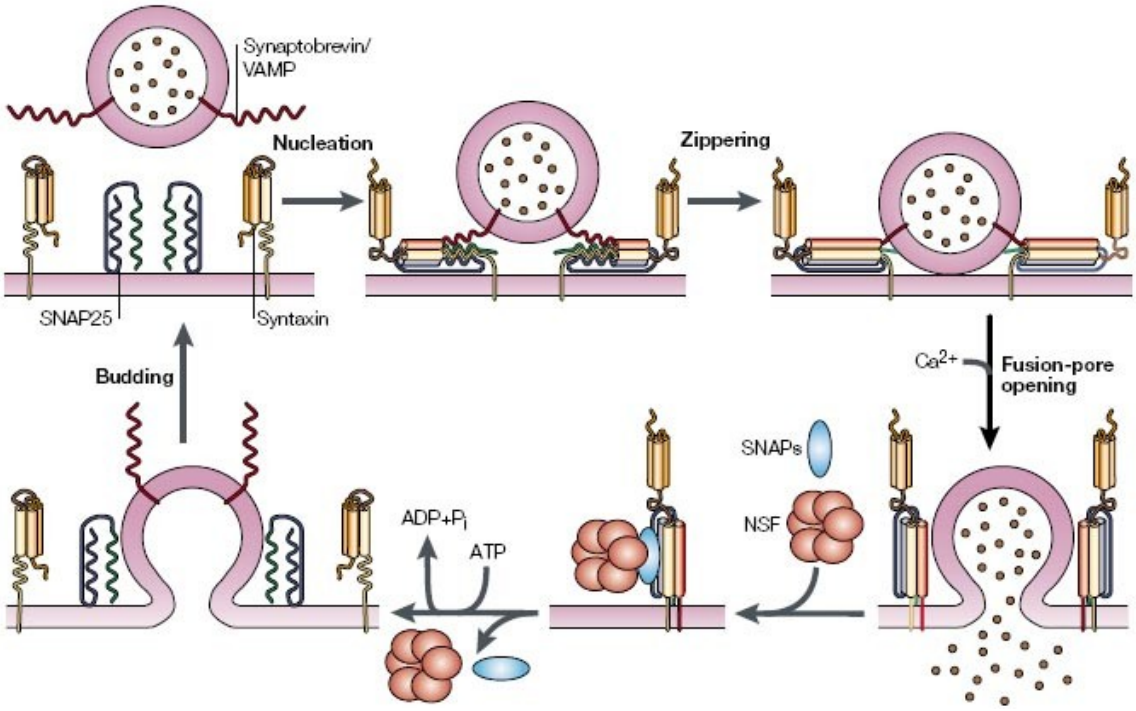


Figure 1.3 The SNARE cycle in synaptic vesicle exocytosis. Upon membrane fusion, the trans-SNARE complex (three of its components are in two opposing bilayers) becomes cis-SNARE complex (all three components are in the same bilayer), and the complex is then unwound in an ATP-dependent process mediated by NSF (*N*-ethylmaleimide-sensitive factor) and SNAPs (soluble NSF attachment proteins). (Fig. 1 of Rizo and Südhof, 2002).

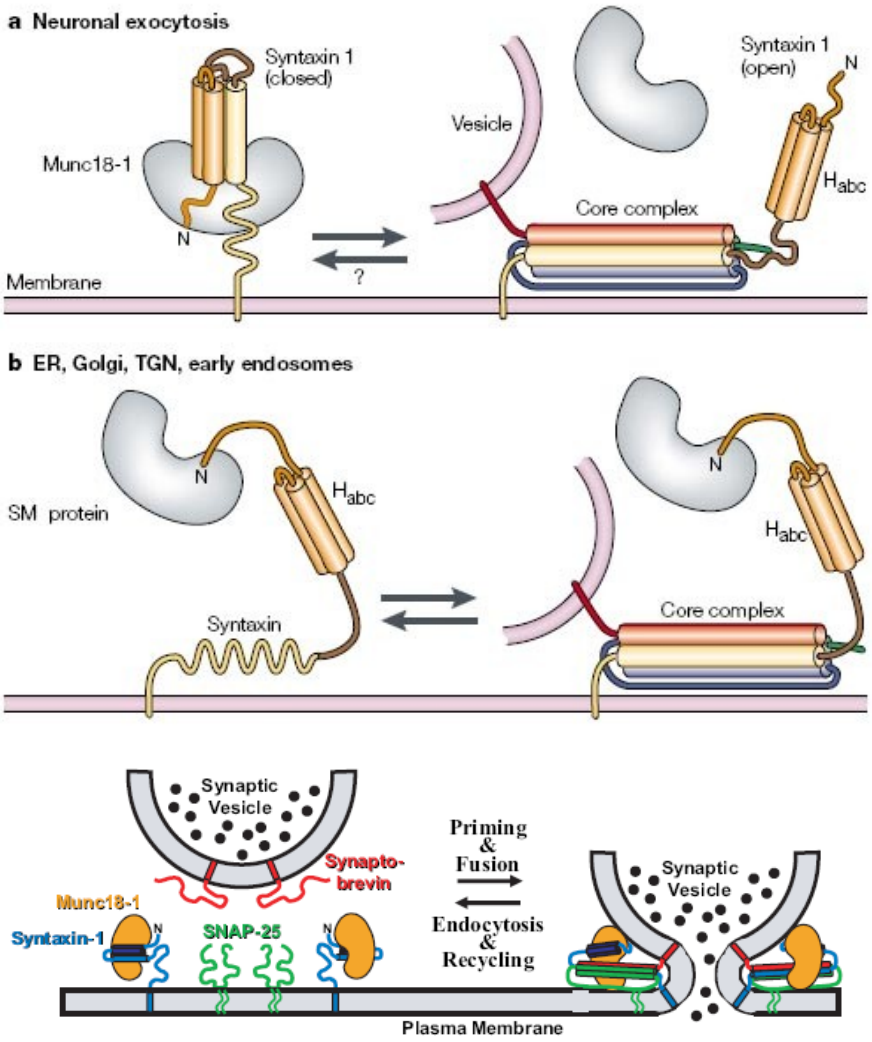


Figure 1.4 Different models of SM protein-syntaxin interaction. Unlike neuronal exocytosis (upper), syntaxin homologues in other vesicular transport systems such as ER and Golgi do not form the closed conformation (middle), and the SM proteins bind to the N-terminus of syntaxin. (Fig. 5 of Rizo and Südhof, 2002). Recently, it was reported that Munc18 does bind to the assembled SNARE complex (lower), and the two different interacting modes (open and closed syntaxin) are essential for synaptic vesicle fusion, and are coupled by functionally critical binding to Stx N-terminus. (Fig. 4 of Dulubova et al., 2007).

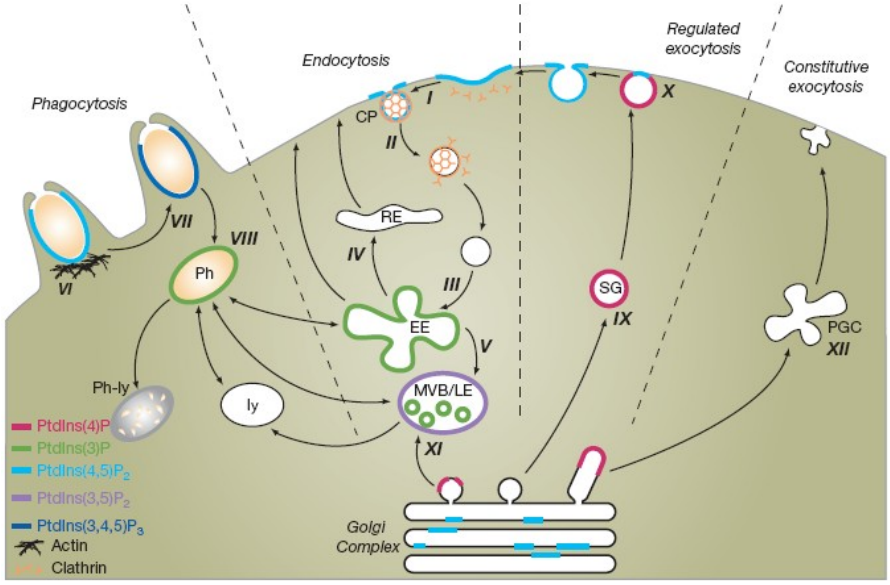


Figure 1.5 PI distribution of endo- and exocytic pathways. CP: coated pit; EE: early endosome; RE: recycling endosome; MVB: multivesicular body; LE: late endosome; ly: lysosome; Ph: phagosome; Ph-ly: phagolysosome; SG: secretory granule; PGC: post-Golgi carrier. (Fig. 2 of De Mattis and Godi, 2004).

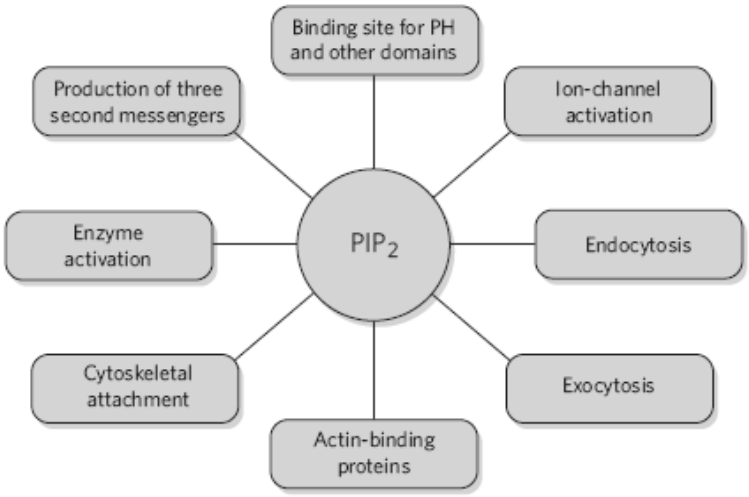


Figure 1.6 Functions of PI(4,5)P2. Phosphatidylinositol 4,5-bisphosphate (PI(4,5)P2) is one of the most versatile PIs that has been implicated in a variety of different physiological activities, including membrane fusion. (Fig. 1 of McLaughlin and Murray, 2005).

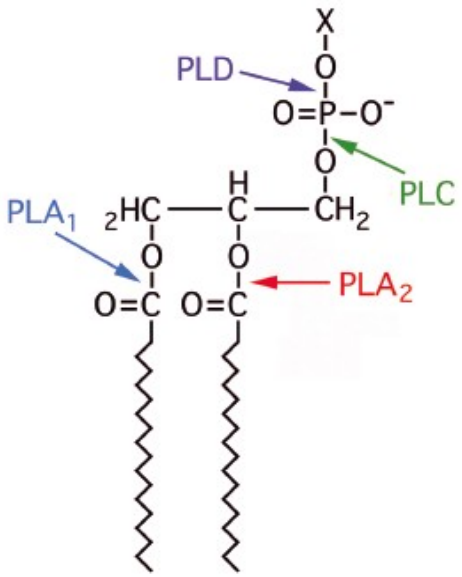


Figure 1.7 Cleavage sites of four classes of phospholipases. Note that the side chains are just for illustration and do not reflect the actual molecular formula. Phospholipase B (not shown) has both PLA_1 and PLA_2 activities. (Fig. 1 of Brown et al., 2003).

Chapter 2. Fusion of artificial liposomes to excised patches

2.1 Introduction

For studying a complex biological system or process, it is always desired to have a functional in vitro system for experimentation. In contrast to the in vivo system, which is complicated by the nature of the biological system itself, an in vitro reconstituted system is a simplified system with all its components defined and controlled. It is complementary to the in vivo system, and provides a convenient and powerful platform for testing novel ideas and hypotheses.

Membrane fusion seems to be a simple biophysical phenomenon, however, it is critical in almost every aspect for an organism to survive, and its regulation is also behind many fundamental biological processes with huge scientific and clinical interests (e.g. neurotransmitter release and information processing, insulin secretion and diabetes, release of cytokines and modulation of the immune system, viral fusion and pathogenesis etc.). Reconstitution of the fusion machinery in vitro “presumably” would allow researchers to manipulate each component in the system, and define the roles of them in an unambiguous way.

Early attempt to reconstitute neurotransmitter release had been done in the early 90's (Karli et al., 1990). Secretory granules (SGs) were purified from bovine chromaffin cells and labelled with fluorescent dye, and membrane vesicles were prepared from bovine adrenal medulla homogenate. In this semi-reconstituted system, fusion of SGs to membrane vesicles was monitored by fluorescent lipid dequenching, and the role of

phospholipase A_2 (PLA_2) was explored (Karli et al., 1990). A totally, and artificially reconstituted fusion system was reported in the late 90's (Weber et al., 1998). Artificial proteoliposomes with defined lipid compositions and v-SNARE (Syb2)/t-SNAREs (SNAP25, Stx1A) were made, and dequenching of fluorescent lipids was also utilized to monitor membrane fusion (Weber et al., 1998). In this reconstituted system, however, the kinetics of membrane fusion was extremely slow, which took hours to reach the plateau, and Ca^{2+} was not required to trigger fusion. In hope to recapitulate the fast Ca^{2+}-dependent membrane fusion, Mahal et al. purified full length Syt1 from bacteria and reconstituted the machinery using the same method (Mahal et al., 2002). Although membrane fusion was stimulated in the presence of Syt1, surprisingly, it was not Ca^{2+}-dependent, and the kinetics was still very slow (Mahal et al., 2002). Later, another group reconstituted Ca^{2+}-dependent membrane fusion in the test tube with similar slow kinetics (Tucker et al., 2004). Rather than using full-length Syt1, they used only C2A-C2B domains from Syt1 (Tucker et al., 2004). They claimed that the Ca^{2+}-independent manner of fusion reported previously (Mahal et al., 2002) was because the bacterial expressed full-length Syt1 was not functional. Ironically, this group also used C2A-C2B domains purified from bacteria (Tucker et al., 2004). The diffusion time required for liposome "docking" was always blamed to be the cause of slow kinetics in this system. However, when liposomes were clustered using streptavidin-biotin-mediated docking system (Schuette et al., 2004), it clearly indicated that the slow kinetics of liposome-liposome fusion was not due to the preceding docking step (Schuette et al., 2004).

In spite of these controversial results, and the debates about the validity of these

experiments (Chen et al., 2006), there are some more creative reconstitution systems made during these years. One of them is the flipped SNAREs system (Hu et al., 2003). The v- and t-SNAREs were inverted and expressed on the plasma membrane of two different cell populations; one with red fluorescent cytoplasm, and one with blue fluorescent nucleus. The two populations were mixed and cell-cell fusion was counted as cells with both fluorescence (Hu et al., 2003). Without commenting on it too much, personally I think the method is cool, and the only advantage is that purification of recombinant proteins is not required in this system.

Another two reconstitution systems monitored fusion by using total internal reflection fluorescence microscopy (TIR-FM). The first one labeled v-SNARE containing vesicles with fluorescent lipids, and then put them onto supporting lipid bilayer containing t-SNAREs. Approaching of the vesicles can be observed as increasing fluorescence intensity, and membrane fusion result in dissipation of the fluorescence due to diffusion of fluorescent lipids in the supporting lipid bilayer (Fix et al., 2004). The other approach prepared the supporting lipid bilayer differently. Instead of making a sheet of lipids on the quartz slide, they coated the slide with NeutrAvidin, and the v-SNARE vesicles containing biotinylated lipids were tethered on the slide without forming a lipid lawn (Yoon et al., 2006). Since t- and v-SNARE vesicles were labeled with two different fluorophores (i.e. green and red), membrane fusion and its intermediates were resolved both by monitoring the change of fluorescence intensity, and the change of FRET efficiency during membrane fusion (Yoon et al., 2006).

Although the TIR-FM approaches are very informative, and the speed of image

acquisition could be very fast, too, inevitably, the temporal resolution of these systems are limited by lipid mixing process, which happens “after” membrane fusion (hemi- or complete) and is restricted by the speed of lipid diffusion. Furthermore, the dilation of the fusion pore could not be monitored, either. From the various methods used to monitor membrane fusion, membrane capacitance measurement gives the highest signal and temporal resolution, and the fusion intermediates can also be resolved. In this project, I was trying to reconstitute the fusion system in giant excised membrane patches, and monitor fusion using capacitance measurement when liposomes containing full-length Syt1 (purified from insect cells) and Syb2 were applied. Unfortunately, after about 1 year of protein purification and half a year of protein reconstitution and capacitance measurements, no convincing liposome fusion was observed. This project was suspended at the end of 2004. The following sections describe the methods, and discuss the results in some more detail.

2.2 Methods

Small-scale plasmid DNA preparation

Everyone likes to use kit to prepare plasmid DNA, but I don't. Unless the DNA is going to be used for transfection, I use DNA prepared by traditional protocol for all other purposes, including DNA sequencing. Miniprep kit is expensive and the eluted DNA concentration is low. To save some money and time for concentrating DNA, I still use the traditional way a lot. The protocol is modified from the standard one, and summarized below.

Single colony of *E. coli* was inoculated into each tube containing 2 ml LB (1% tryptone, 0.5% yeast extract, 1% NaCl) medium with appropriate antibiotic, and cultured at 37°C overnight with vigorously shaking. The overnight culture was poured into a 1.5 ml microtube and centrifuged at 12000 g for 30 s. The supernatant was removed and the pellet was resuspended in 100 μl Solution I (50 mM glucose, 25 mM Tris-Cl, 10 mM EDTA, pH 8.0). An amount of 200 μl freshly prepared Solution II (0.2 N NaOH, 1% SDS) was added into the suspension, and the microtube was inverted gently for several times to lyse the bacteria. Subsequently, 150 μl ice-cold Solution III (3 M potassium acetate, 11.5% glacial acetic acid) was added and the mixture was inverted gently for several times and placed on ice for 3-5 minutes (optional). After centrifugation at 12000 g for 5 minutes, the supernatant containing plasmid DNA was transferred into a fresh microtube.

Phenol/chloroform extraction is optional. If desired, an equal volume of

phenol:chloroform:isoamyl alcohol (25:24:1) was added and mixed thoroughly with the crude plasmid DNA, and then centrifuged at 12000 g for 2 minutes. The aqueous layer (upper portion) of the mixture was transferred to a new tube and the plasmid was precipitated by adding 1/10 volume of 3M sodium acetate, pH 5.2 (optional), and 2 volumes of anhydrous ethanol. If higher yield was needed, the mixture was kept at -20°C for 20 minutes, else it was subjected to centrifugation directly. Centrifugation was performed at 12000 g for 20 min at 4°C, and the supernatant was discarded. The plasmid pellet was washed in 1 ml of 70% ethanol (optional) and centrifuged at 12000 g again for 5 minutes. The supernatant was discarded and the pellet was dried either by air or vacuum. The plasmid was dissolved in desired volume of either double distilled water or TE buffer (10 mM Tris-Cl, 1 mM EDTA, pH 8.0). RNA was removed by RNaseA digestion before DNA sequencing, no further purification was performed.

SDS-PAGE analysis

The mini-gel casting stand and frame (Bio-Rad) were assembled, and all glass plates were cleaned and dried with ethanol before use. For a single 10% separating gel, the following ingredients were mixed: 1.5 ml water, 1.5 ml of 50% glycerol, 1.85 ml of 1.5 M Tris-Cl buffer (pH8.8), 75 µl of 10% SDS, 2.5 ml of 30% acrylamide/0.8% bisacrylamide, 55 µl of 10% ammonium persulfate, and 5.5 µl of TEMED (*N*,*N*,*N*,*N*-tetramethyl-ethylenediamine). The mixture was poured into the casting module, and water-saturated isobutyl alcohol was added to the top of the gel. The gel was polymerized at room temperature for about 30 min, and the isobutyl alcohol was then poured off from the gel. The residual alcohol was removed using paper towel. For the 3% stacking gel, the

followings were mixed: 2.375 ml water, 0.94 ml of 0.5 M Tris-Cl buffer (pH6.8), 37.5 μl of 10% SDS, 0.375 ml of 30% acrylamide/0.8% bisacrylamide, 55 μl of 10% ammonium persulfate, and 5.5 μl of TEMED. The mixture was poured onto the separating gel, and the comb was then inserted into the stacking gel. The stacking gel was polymerized for additional 30 min.

The electrophoresis module and cell were assembled, and filled up with tank buffer (for 1 L, add 3 g Tris base, 14.4 g glycine, 1g SDS to water). Equal volume of 2X sample buffer (for 10 ml, mix 4 ml of 10% SDS, 3 ml of 0.5 M Tris-Cl pH6.8, 2 ml of 50% glycerol, 1 ml β-mercaptoethanol, and some bromophenol blue) was mixed with desired amount of protein sample, and the mixture was heated in boiling water for ~3 min. The boiled samples were placed on ice for few minutes before loading. Electrophoresis was carried out at 90 V before the dye was stacked on top of the separating gel, after that, the voltage was increased to 120 V.

After electrophoresis, the module was disassembled, the stacking gel was discarded, and the separating gel was subjected to Coomassie blue staining or western blot. For Coomassie blue staining, the gel was immersed in staining solution (45% methanol, 10% acetic acid, 0.05% Coomassie brilliant blue, reusable) for ~2 hr and then destained in buffer composed of 30% methanol and 10% acetic acid for several times. After destaining, the gel was immersed in solution composed of 35% ethanol and 2% glycerol for 2-3 hr, and then dried and framed in cellophane sheets for record.

Western blot

Nitrocellulose membrane was used blotting. To make the transfer sandwich, the followings were stacked in an order of anode to cathode: a sponge pad, 2 sheets of Whatman no. 1 filter paper, nitrocellulose membrane, gel, 2 sheets of filter paper, and a sponge pad. The transfer sandwich was immersed in cold transfer buffer (for 1 L, add 3 g of Tris base, 14.4 g glycine, and 200 ml methanol to water, reusable), and the transfer was carried out with a voltage of 60 V for 5 hr at 4°C.

After transfer, the transfer sandwich was disassembled, and the membrane was placed in blocking buffer (10% skim milk prepared from powder, in TBST pH 8.0 solution which contains 10 mM Tris-Cl, 150 mM NaCl, and 0.05% Tween 20) at room temperature for 0.5-1 hr with shaking. The membrane was rinsed with TBST, and then incubated with primary antibody which was diluted in 5% skim milk/TBST solution for 1 hr at room temperature. After washing with TBST for 3X5 min, the membrane was incubated with horseradish peroxidase (HRP)-conjugated secondary antibody for another 1 hr at room temperature. Subsequently, the membrane was washed again with TBST for 3X10 min with vigorous shaking.

For chemiluminescent detection, the homemade ECL solutions were used. The ECL solution A contains 5 mM luminol, 200 μM *p*-coumaric acid, and 100 mM Tris-Cl pH8.5, protected from light. The ECL solution B contains 5.4 mM H_2O_2 (from 30% concentrate), and 100 mM Tris-Cl pH 8.5. In my experience, the quality of H_2O_2 is critical for the quality of the prepared ECL solutions, and I use H_2O_2 from Fluka.

Equal volume of the two solutions were mixed right before use, and dispersed evenly

on the membrane for ~1 min. The excess of ECL mixture was removed by blotting the membrane with task wipers (Kimwipes), and the exposure and development of the film was performed in a darkroom.

Dot blot – *Express*

For chromatographic experiments with sample of unknown elution volume, it might be impractical to run standard western blot for each collected fraction to determine fractions used for the next column. To speed up the process, a western blot without the gel-running and transfer steps was performed.

Two microliter of 2X sample buffer and 2 µl sample from each fraction were mixed, and 3 µl of each boiled sample were dotted on a nitrocellulose membrane. To place the dots, the nitrocellulose membrane was placed on a paper towel before dotting to avoid excessive spreading of the samples, and the plastic insert from a 200 µl EZ Rack tip transfer system (Denville Scientific) was placed on top of the membrane as a dotting guide. Dots were dotted through the holes of the plastic insert. Since time, but not quality of the blot was the major concern here, the lengths of subsequent blocking, washing, and incubation were all shortened.

Purification of Synaptotagmin 1 from Sf9 cells

The Syt1 expression construct contains coding sequences of Syt1 fused with honeybee melittin secretory signal (HBM) and His tag (eight histidine) sequences on its N-terminus. The HBM-$(His)_8$-Syt1 fragment was generated via PCR and subcloned into pCR2.1-TOPO vector (Invitrogen). After sequence confirmation, the fragment was cut

out by *Nco*I, Klenow treatments followed by *Eco*RI digestion, and then cloned into *Eco*RI,-*Stu*I site of pFastBac vector (Invitrogen). The resulting plasmid (pFB-hhSyt1) was transformed into DH10Bac competent cells (Invitrogen), which contain parent bacmid and helper plasmid for transposition. Gentamicin resistant gene and fusion gene driven by baculovirus polyhedrin promoter on pFB-hhSyt1 were transposed into the parent bacmid, and the transposition was verified using PCR. To produce recombinant baculovirus, Sf9 cells were plated on a 6-well cell culture plate, and the recombinant bacmid was transfected into Sf9 cells using Cellfectin (Invitrogen) according to the manufacture's instruction. The P1 recombinant viruses were harvested from the medium for further amplification, and the Sf9 cells were scraped from the plate for western blot to confirm the production of recombinant protein. For short-term storage, 2% FBS was added into medium containing recombinant viruses, kept at 4°C, and protected from light. For long-term storage, the viral stock was stored at -80°C.

For construct containing Syt1 and His-tag fused to its C-terminus, Syt1-$(His)_8$ fragment was also generated by PCR, cloned into pCR2.1-TOPO vector, and then subcloned into pFastBac vector for producing recombinant bacmid. The fragment was excised from pCR2.1-TOPO by *Nco*I digestioin, Klenow treatments, and then by *Hin*dIII digestion. It is cloned into *Hin*dIII,-*Stu*I site of pFastBac vector. The procedures for producing recombinant baculovirus were the same as described above.

For large-scale expression of Syt1 in Sf9 cells, 40 ml recombinant viruses were added into 1.6 L of Sf9 cells at a density of ~$2X10^6$ cells/ml. Cells were cultured at 27°C with vigorous shaking and harvested after 2 days. The harvested cells were broken using one

freeze-thaw cycle followed by several times of sonication. The lysate was subjected to centrifugation using a JA20 rotor (Beckman Coulter) at 15 krpm for 15 min, and the membrane pellet was resuspended in extraction buffer (50 NaH_2PO_4, 300 NaCl, 2 β-mercaptoethanol, 1 PMSF, 2 imidazole, in mM, and supplied with 2% Triton X-100). Extraction was carried out at 4°C for ~1 hr with gentle agitation. After extraction, membrane debris were removed by centrifugation at 15 krpm for 15 min with a JA20 rotor, followed by additional centrifugation at 35 krpm for 30 min with a Ti70 rotor (Beckman Coulter). The supernatant was collected into a 50 ml centrifuge tube, and 2.4 ml of Ni-NTA beads and additional 1 mM of PMSF were added into the crude protein lysate.

After overnight binding at 4°C with gentle agitation, the beads were washed sequentially with 12 ml of 5, 10, 20 and 30 mM imidazole prepared in solution containing Triton X-100 (50 NaH_2PO_4, 300 NaCl, 2 β-mercaptoethanol, 1 PMSF, in mM, and supplied with 0.1% Triton X-100), and then washed with 40 and 50 mM imidazole prepared in the same solution except that 0.1% Triton X-100 was substituted by 1% octyl β-D-glucopyranoside (OG). To elute the protein, 2.5 ml of 100, 150, 200, 250, and 500 mM imidazole (prepared in the same OG solution) were added sequentially, and the eluates were collected in 1.5 ml microtubes. Quality of each fraction was examined using Coomassie blue staining after electrophoresis. Molar concentration of Syt1 was determined using dot blot with serially diluted samples, standard (C2AB of Syt1 with known concentation), and monoclonal antibody CL41.1 (Synaptic Systems).

Purification of Synaptobrevin 2 from *E. coli*.

The protocol for purifying Syb2 was adopted from protocol described previously (Chen et al., 2006). The GST (glutathione S-transferase)-Syb2 fusion construct was transformed into *E. coli*. strain BL21. A single colony of the transformed bacteria was inoculated into 100 ml of YTA medium (1.6% tryptone, 1% yeast extract, 0.5% NaCl), and cultured at 37°C overnight with vigorous shaking. The overnight culture was added into 2 L of fresh YTA medium and further cultured at 37°C for several hours until the OD_{600} reached ~0.8. Induction of Syb2 expression was done by adding IPTG (isopropyl β-D-1-thiogalactopyranoside) to a final concentration of 0.5 mM. Cells were then culture at room temperature for additional ~6 hr with vigorous shaking before harvest.

The cells were spun down, rinsed with PBS (prepared from PBS tablet, Sigma), and then resuspended in extraction buffer containing PBS, 2 mM EDTA, 1 mM EGTA, 0.05% Tween 20, 0.4% Triton X-100, 0.5% *N*-lauroylsarcosine, 25 μg/ml lysozyme, 2 mM β-mercaptoethanol, and protease inhibitors. Cells were broken in extraction buffer by sonication, and then kept at 4°C for 30 min with gentle agitation to extract membrane proteins. Membrane debris were removed by centrifugation at 15 krpm for 10 min with a JA20 rotor, followed by additional centrifugation at 35 krpm for 30 min with a Ti70 rotor. The supernatant was collected into a 50 ml centrifuge tube, and 3 ml of glutathione sepharose 4B beads and additional 1 mM of PMSF were added into the crude protein lysate.

After overnight binding at 4°C with gentle agitation, the beads were washed with ~100 ml PBS with added 0.1% Triton X-100 and 2 mM β-mercaptoethanol. To remove the

GST tag, the beads were resuspended in 1.5 ml of the same buffer containing ~2 units of thrombin and kept at room temperature for 4 hr with gentle agitation. After cleavage, the supernatant was collected by centrifugation at 500 g for 5 min, and the beads were rinsed again using the same buffer. The 0.5 ml supernatant were combined with the previous one (~2 ml), and then mixed with 20 ml of Mono S loading buffer (50 mM NaCl, 25 mM HEPES, 0.1% Triton X-100, 2 mM β-mercaptoethanol, pH7.0).

For Mono S chromatography, the column was equilibrated with the same buffer described above before the sample was loaded. After loading, the column was washed with 15 ml of buffer containing 100 mM NaCl, and Syb2 was eluted by ramping the NaCl concentration gradually from 100 mM to 400 mM in a total volume of 20 ml. The elution peak of Syb2 was located at a NaCl concentration of 150-200 mM. The concentration of the eluted Syb2 was determined by measuring OD_{280}.

Reconstitution of Syt1 and Syb2 into liposomes

The detergent-mediated direct incorporation method (Rigaud et al., 1995; Rigaud and Lévy, 2003) was used for Syt1/Syb2 reconstitution. The protocol for preparing large unilamellar vesicles (LUVs) is from Avanti polor lipids (Alabaster, AL). To prepare LUVs, POPC (1-palmitoyl-2-oleoyl-*sn*-glycero-3-phosphocholine) and DOPS (1,2-dioleoyl-*sn*-glycero-3-phosphoserine) were mixed in a glass tube with a molar ratio of 85:15. The solvent (chloroform) was expelled from the lipids under N_2 flow, and the lipid film was further dried under vacuum for an additional hour. An appropriate amount of buffer (buffer for electrophysiological recording) was added into the glass tube to make

the final lipid concentration 15 mM, and the tube was agitated strongly using Vortex for at least 5 min to hydrate the lipid film. The hydrated lipids were moved to a 15 ml centrifuge tube, and subjected to 5 freeze/thaw cycles using liquid nitrogen and warm water to increase entrapment of water-soluble compounds. The lipids formed large multilamellar vesicles (LMVs) after these steps.

The extrusion method was used to make LUVs from LMVs. The mini-extruder was assembled according to the manual (Avanti polar lipids), and a polycarbonate membrane with 80 nm pore size was employed. After several times of extrusion, the LUVs were collected in a 1.5 ml microtube and stored at 4°C. The diameter of LUVs made by this method is about 100 nm.

To make Syt1/Syb2 proteoliposomes, 1 volume of liposomes (total lipid concentration 15 mM) was mixed with desired amount of Syt1 and Syb2 recombinant proteins (contains 1% OG). The total volume was adjusted to 3 volumes, and the OG concentration was adjusted to 0.77%. Note that if buffer needs to be added into the mixture, add buffer prior to adding proteins because LUVs may be solubilize if the OG concentration (which is present with proteins) is too high. After mixing, the total lipid concentration of the mixture should be 5 mM. The mixture was kept at room temperature for 30 min for protein incorporation.

Subsequent dialysis was used to remove OG from proteoliposomes. The liposomes were dialyzed against 500 ml of electrophysiological recording buffer at room temperature without stirring for 30 min, and then dialyzed against 500 ml and 1 L buffer at room temperature with stirring for 30 min and 1 h, respectively. The final dialysis step

was carried out at 4°C overnight with 1 L buffer and 1 g of Bio-Beads (Bio-Rad) added to the buffer to absorb residual OG. The detergent-free proteoliposomes were stored at 4°C.

Quality assays of the reconstituted proteoliposomes

The size distribution of proteoliposomes was measured by using dynamic light scattering. For leakage assay, LUVs containing self-quenched carboxyfluorescein were made. The lipid film was hydrated with buffer containing 100 mM 5(6)-carboxyfluorescein, and proteoliposomes were made using the same procedures as described above. The proteoliposomes were diluted 80X in buffer, and the leakage of the content (i.e. carboxyfluorescein) was measured by means of dequenching of leaked carboxyfluorescein in a fluorescence spectrophotometer. At the end of measurement, liposomes were lysed by adding 1% Triton X-100 to the cuvette to get maximal fluorescence.

To test if recombinant proteins were incorporated into liposomes rather than sticking on them, urea (4M stock prepared in 40 mM MES, pH6.5) was added to proteoliposomes in a final concentration of 2 M to remove proteins that were not incorporated (Schwab et al., 2000). After 30 min extraction at 4°C, the mixture was centrifuged at 65 krmp for 1 hr with a TLA-120.1 rotor (Beckman Coulter). The supernatant was transferred to a new microtube, and the pellet containing incorporated proteins was resuspended in equal volume (volume of the supernatant) of buffer. Dot blot with serial diluted samples was applied to examine the incorporation.

Orientation test was done by treating proteoliposomes with proteases followed by

western blot with designated antibodies. To test the orientation of Syt1, trypsin and antibodies V216 (against cytoplasmic side of Syt1), V761 (against lumen portion of Syt1) were used. For Syb2, chymotrypsin and polyclonal antibody P939 were employed. For control experiment, proteoliposomes were lysed by adding Triton X-100 to expose all the proteins before enzymatic digestion. Western blot was used simply because the protein concentration was not high enough to be seen using Coomassie blue staining.

Triple-marker expression system for Stx1A, SNAP25A, and Munc18-1

Three bicistronic cloning vectors were engineered. They are pBNG2, pBMG, and pBPR, which express nucleus-localized EGFP, membrane-targeted EGFP, and peroxisome-localized DsRed, respectively. When three of them are expressed in the same cell, the cell has green nucleus, green membrane, and red puncta in the cytoplasm which serve as markers for co-expression of proteins of interests. Theoretically, two different colors (e.g. green and red) can be targeted to these three cellular compartments (i.e. nucleus, plasma membrane, and cytoplasm) using this approach, and cells expressing a maximum of 6 kinds of different proteins can be distinguished from others. They have yellow (if green and red colors are used) nucleus, membrane, and yellow puncta in the cytoplasm.

The backbone vector is pIRES2-EGFP (Clontech), and an *Age*I site was introduced between the IRES (internal ribosomal entry site) and the fluorescent protein coding sequence. The *Age*I site was used for later cloning steps. To make the backbone vector, pIRES2EGFP was digested by *Bst*XI, followed by Klenow fill-in, and *Not*I digestion. The gel-extracted vector was then ligated with EGFP sequence, which was obtained from

pEGFP-N1 (Clontech) by treating them with *Bam*HI, Klenow fill-in, and *Not*I digestion. The resulting vectors are pIRES-AgeI-EGFP.

To make pBNG2 (bicistronic, nucleus is green), three tandem repeats of nuclear localization signal (NLS) from SV40 large T-antigen (Fanara et al., 2000) fused with EGFP coding sequence were generated via PCR, and then subcloned into pCR2.1-TOPO vector. After sequence confirmation, the 3NLS-EGFP fragment was excised from the pCR2.1TOPO vector by *Age*I and *Not*I digestion, followed by gel extraction. Vector pIRES-AgeI-EGFP was treated the same way, and ligated with the 3NLS-EGFP fragment (*Not*I is at the 3'-end of EGFP).

For pBMG (bicistronic, membrane is green), the same strategy was employed. Fragment containing coding sequence of membrane-targeting domain of Src oncoprotein (Sigal et al., 1994) fused to the 5'-end of EGFP coding sequence was generated by PCR, and cloned into pCR2.1TOPO vector. The fragment was then subcloned into *Age*I-*Not*I site of pIRES-AgeI-EGFP.

To construct pBPR (bicistronic, peroxisomes are red), coding sequence of peroxisomal targeting sequence 1 (PTS1, Clontech) was fused to the 3'-end of DsRed2 (Clontech) coding sequence. The DsRed2-PTS1 fragment was generated via PCR and cloned into pCR2.1TOPO vector. After sequence confirmation, the DsRed2-PTS1 fragment was excised from the pCR2.1TOPO vector by *Eco*RI digestion, followed by Klenow fill-in and *Not*I cleavage. Vector pIRES-AgeI-EGFP was treated by *Bst*XI digestion and Klenow fill-in, followed by *Not*I cleavage, and then ligated with the DsRed2-PTS1 fragment.

Rat Stx1A was cloned into *Sma*I-*Eco*RI site of pBNG2 to generate pBNG2-Stx1A. Human SNAP25A was cloned into *Sma*I-*Pst*I site of pBNG2 and *Sma*I-*Pst*I site of pBMG to generate pBNG2-SNAP25A and pBMG-SNAP25A, respectively. For rat Munc18-1, it was cloned into *Sma*I-*Eco*RI site of pBPR to generate pBPR-Munc18.

Liposome perfusion and capacitance measurement in excised patches

Since there was only a limited amount of liposomes, special perfusion system were made. Two quartz tubings with 40 μm inner diameter were stuck together to make a mini-θ tube. The mini-θ tube was fixed into a glass pipette, and the orientation of the mini-tube was marked on the glass pipette under stereomicroscope. The perfusion system was then mounted to the microscope, and positioned according to the marker on the glass pipette. If liposomes were not loaded with carboxyfluorescein, a tiny amount of carboxyfluorescein was mixed with liposomes prior to experiments so that solution flow could be monitored under fluorescence microscope. An amount of 30 μl liposomes was enough to use for an entire day, and air pressure from a syringe was used to control the solution flow as used in the intra-pipette infusion system. Temperature was set to ~37°C by a heating fan with its sensor stuck on the stage of the microscope.

Patches were excised from INS-1 cells or cells co-expressing Stx1A, SNAP25A, and Munc18-1. Two different protocols for triggering liposome fusion were tested. The first one mixed 1mM Ca^{2+} with liposomes first, and then moved the patch from blank liposomes/Ca^{2+} to proteoliposomes/Ca^{2+}. In the second protocol, patches were placed under blank liposome/Ca^{2+} to trigger endogenous vesicles fusion, and then incubated with

Syt1/Syb2 proteoliposomes for 1 min. The patches were then moved back to blank liposomes/Ca^{2+} again to trigger fusion. This protocol was referred to as the docking/priming protocol.

2.3 Results and discussion

Purification of Synaptotagmin 1

Before the HBM-$(His)_8$-Syt1 and Syt1-$(His)_8$ fusion constructs were made, several other expression constructs were used for expressing Syt1. One of them co-expressed Syt1 and Syb2 in Sf9 cells (made by Jiong Tang in Dr. Südhof's lab, UTSouthwestern), and there was no affinity tag fused with Syt1 coding sequence. To purify the untagged Syt1, several chromatographic columns were employed sequentially. They were DEAE (diethylaminoethyl), CM (carboxymethyl), HA (hydroxyapatite), and Mono S (Amersham) columns. Pioneer experiments were done to determine the fractions subjected to the next column, and the conductance readings of the fractions were used as references. Dot blot using anti-Syt1 antibody and representative fractions from each column are shown in Fig. 2.1A. The HA column was able to separate Syt1 monomer, dimer, and degraded Syt1 (fractions 11, 18, 14 in Fig. 2.1B). After further purification using Mono S column, the concentration of the most concentrated fraction was only ~0.15 μg/ml. This concentration is 1000 times lower than the practical concentration for reconstitution, and the quality is not good, either (not shown).

In order to increase the quantity and quality of the purified Syt1, a $(His)_6$ tag was added to the N-terminus of Syt1. However, Coomassie blue staining of the affinity purified Syt1 indicated that the quality was still poor, and the concentration was only about 0.5-1 μg/ml (Fig. 2.2). Two other expression constructs were made. One is Syt-$(His)_8$, and the other one is HBM-$(His)_8$-Syt1. The number of histidine was increased in

hope that the binding affinity would be higher, and more stringent washing conditions could be applied. For Syt-$(His)_8$, the tag was moved to the C-terminus simply because the N-terminus-tagged construct did not express well. For HBM-$(His)_8$-Syt1, the honeybee secretory signal was added at the very N-terminus to ensure that the mammalian type I transmembrane protein (e.g. Syt1) can be inserted into the membrane correctly in insect cells, and thus increase the final yield.

As shown in Fig. 2.3, the quality and quantity of both expression constructs were significantly increased (although still not perfect). Using dot blot and serial dilution, the concentration of the most concentrated fraction was ~300 μg/ml C2AB domain for both constructs (Fig. 2.4). In large-scale preparation, the protein expression level of HBM-$(His)_8$-Syt1 was much higher than that of Syt-$(His)_8$ (~15 mg v.s. ~4 mg C2AB domain), however, the final yield was similar (Fig. 2.4). More Ni-NTA beads should be added into the crude protein lysate of HBM-$(His)_8$-Syt1 in future experiments because about half of the expressed protein was in the flow-through (Fig. 2.4). In addition, one more chromatographic purification step may be desired to further purify the product.

Purification of Synaptobrevin 2

After affinity and ion exchange chromatographic purification, the quality of the purified Syb2 was quite good (Fig. 2.5). The concentration of the most concentrated fraction was ~450 μg/ml. Although the quality was good, and the quantity was enough for reconstitution, it is possible to further optimize the expression and purification procedures. For example, incubating the cells at 37°C after IPTG induction may increase

the total expressed Syb2 tremendously (personal communication, Chen et al., 2006). In addition, the protocol for chromatographic purification may also need to be adjusted since there was still a lot of Syb2 in the flow-through (Fig. 2.5).

Quality of the liposomes

Syb2 reconstitution with detergent-mediated direct incorporation method has been well documented (Chen et al., 2006), however, reconstitution of integral Syt1 using this method has never been published as of July 2004. To test the right condition for Syt1 reconstitution, different OG concentration were tested. OG concentration of 0.67% was used for Syb2 proteoliposome preparation. However, as shown in Fig. 2.6 (columns 5-7), it is clear that Syt1 tends to form aggregation at this concentration, and the best OG concentration for Syt1 reconstitution is 0.77% (if total lipid concentration is 5 mM). To check if the "reconstituted" proteins really get integrated into liposomes, 2 M urea was used to remove proteins sticking on the liposomes. Again, reconstitution condition for Syb2 (Fig. 2.6 column 4) is not suitable for reconstituting Syt1 (Fig. 2.6 column 1), and OG concentration of 0.77% is still the best for Syt1 reconstitution (Fig. 2.6 column 2). This OG concentration was selected for subsequent Syt1/Syb2 reconstitution.

There was a reason to pick these three OG concentrations, I did not choose them randomly. The monomeric OG concentration in water is 17 mM (Rigaud et al., 1995). The OG-lipid molar ratio for OG-saturated liposomes is 1.3, and when liposomes are totally solubilized, the ratio is 3 (Rigaud et al., 1995). That is, when liposomes with 5 mM of total lipids are saturated, the OG concentration is,

$17+5\times1.3=23.5$ (mM).

When liposomes are totally solubilized, the concentration is,

$17+5\times2.6=30$ (mM).

The molecular weight of OG is 292.4, so 23.5 and 30 mM OG represent 0.69% and 0.88% OG, respectively. OG concentration of 0.77% is between these two values.

The size distribution of reconstituted liposome is shown Fig. 2.7. Distribution of LUVs made by extrusion method was very homogeneous, and the diameter was about 100 nm (upper panel). When Syt1 was reconstituted using 0.67% OG (previously established OG concentration for reconstituting Syb2, middle panel), the size distribution was very heterogeneous, and there seemed to be particles with large diameter which was consistent with previous incorporation assay (Fig. 2.6 column 5). Using OG concentration of 0.77%, the size distribution of Syt1-containing proteoliposomes was improved tremendously (Fig. 2.7, lower panel), although it was still not as homogeneous as Syb2-only proteoliposomes (Chen et al., 2006). Syt1-containing proteoliposomes (reconstituted using 0.77% OG) did not show significant sign of leakage within a two-hour period (Fig. 2.8).

The orientations of incorporated proteins were tested by enzymatic digestion followed by western blot with antibodies against cytoplasmic or lumen portion of the proteins. With antibody against Syt1 lumen portion (V761), the smear on western film indicated correct-orientated Syt1 (Fig. 2.9 A). As shown in the boxed region, there were probably more than 70% of Syt1 were inserted in the correct orientation. To further confirm the

result, antibody against Syt1 cytoplasmic portion (V216) was employed. Signals remained on the film indicated inversely-incorporated Syt1 (Fig. 2.9 B). As indicated in the boxed region, there were probably less than 30% of Syt1(compared with total input) that were in inverted orientation when reconstituted in this fashion. The orientations of Syb2 in Syt1/Syb2 (prepared in 0.77% OG) and Syb2-only (prepared in 0.67% OG) proteoliposomes were also tested (Fig. 2.9 C). The ratios of correctly incorporated Syb2 were similar in both cases. According to this blot (Fig. 2.9C), probably ~60% of Syb2 were in correct orientation. Note that the bands in this blot were saturated, so the estimation might not be accurate. According to Chen et al. (2006), there are more than 80-90% of Syb2 incorporated correctly in Syb2-only liposomes when 0.67% OG were used.

In conclusion, to reconstitute Syt1 and Syb2 into liposomes at the same time, the detergent-mediated direct incorporation method with 5 mM total lipids and 0.77% OG gives proteoliposomes with satisfactory quality, in the aspects of size distribution, leakage of content, protein incorporation, and orientation.

Establishment of the triple-marker transient expression system

The original plan was to excise giant patches from cells transiently expressing Stx1A, SNAP25A and Munc18-1, and then perfuse Syt1/Syb2 proteoliposomes to the cytoplasmic side of the membrane. One immediate technical challenge was to identify cells co-expressing all three proteins under regular epifluorescence microscope. It is common to locate cells co-expressing 2 different proteins using green and red fluorescence as markers, and it is also possible to use a 3rd color as a marker for the 3rd

protein of interests. However, identifying cells with correct color mixture among other fluorescent cells is also quite challenging, and the microscope also needs to be equipped with different excitation light sources and filter sets, which is very unusual for a microscope used for patch clamping. To overcome this limitation, I designed a marker system that also uses the localizations (i.e. nucleus, membrane, and cytoplasm) of fluorescent proteins as markers. Theoretically, two different colors (e.g. green and red) can be targeted to these three cellular compartments, and cells expressing a maximum of 6 different proteins can be distinguished from others using the combination of color-localization information. I further targeted cytoplasmic fluorescent proteins to peroxisome to ensure that membrane-targeted fluorescent proteins can be distinguished from them. Ideally, cells co-expressing Stx1A, SNAP25A and Munc18-1 look like Fig. 2.10A.

The idea was great, but the outcome was sad. The major problem was that the membrane-targeted EGFP could not been distinguished clearly from nucleus-targeted EGFP under epifluorescence microscope (Fig. 2.10 B, C). I am sure that they can be distinguished under confocal microscope, however, I do not have a confocal with my patch clamp setup, and it is also impractical to scan every cell to locate the right one. Another problem was the transfection efficiency. Even when the transfection efficiency looked great on the culture dish, it was still not easy to find a cell that expresses two different colors after trypsinization. Presumably because the transfection efficiency was over-estimated when the cells were still attached to the culture dish.

As a last resort, I started to make cell lines co-expressing all three components.

Expression constructs pBNG2-Stx1A, pBPR-Munc18, and pBNG2-SNAP25A were used. I did not use pBMG-SNAP25A simply because at the time of transfection, I did not have kit-purified pBMG-SNAP25A by my hands. After transfection, HEK293 cells were diluted and plated onto cell culture dishes. Mediums with 600, 400, and 200 μg/ml G418 were applied to select cells containing expression vectors. Single colonies of cells were subcloned, and western blot with antibodies against Stx1A, SNAP25A, and Munc18-1 was employed to check expression pattern. As shown in Fig. 2.11, clones 16, 17, and 19 contain all three components.

Perfusion of artificial liposomes to excised patches

To make a long story short, I did not observe any convincing liposome fusion, and this project was suspended. However, there were still something to be reported, or discussed.

It is critical to saturate the pipette with liposomes before perfusing proteoliposomes, because capacitance increases and gets saturated even when blank liposomes without Ca^{2+} are perfused (not shown). There are two explanations, one is that the increase of capacitance is caused by fusion of blank liposomes to the patch; the other one is that lipids stick onto the glass pipette and then diffuse into the patch membrane. The answer is still unknown, and amperometry is probably required to answer this question.

Anyway, after saturating the pipette with blank liposomes/Ca^{2+}, it appeared that the speed of capacitance increment became faster after moving the patch to Syt1/Syb2 proteoliposomes in the presence of Ca^{2+} (Fig. 2.12 A). It seemed that I have got the first electrophysiological recording of Ca^{2+}-dependent liposome fusion. However, when the

docking/priming protocol was applied, I could not see any capacitance jump upon Ca^{2+} application (not shown). More importantly, capacitance started to increase when Syt1/Syb2 proteoliposomes were applied in the absence of Ca^{2+} (not shown). This observation implies that either the fusion of proteoliposomes is Ca^{2+}-independent (Mahal et al., 2002), or the proteoliposomes stick onto the glass pipette faster than blank liposomes do. To test it, I saturated the pipette with proteoliposomes without Ca^{2+} first, and then move the patch to proteoliposomes with Ca^{2+} to see if faster capacitance increment (as Fig. 2.12 A) could be observed. Unfortunately, there was nothing happening when Syt1/Syb2 liposomes were applied with Ca^{2+} (Fig. 2.12 B), indicating that the previous result could be an artifact.

There are too many unknowns in this novel system. For example, if blank liposomes could fuse to the membrane or just stick onto the pipette is unknown, and I also could not exclude the possibilities that some of the proteoliposomes might fuse to the membrane in a Ca^{2+}-independent way. Amperometry with liposomes loaded with dopamine or serotonin is probably required to answer these questions. Sadly, time was passing and I could not establish the amperometric recording setup before this project was suspended.

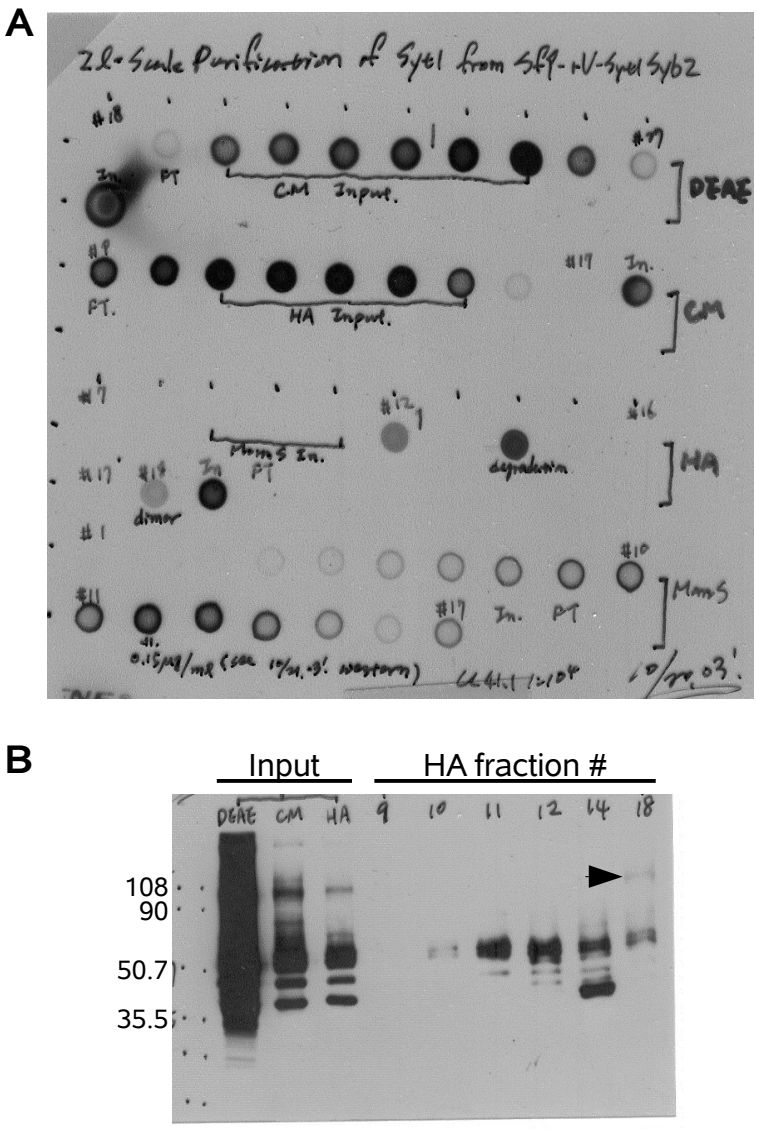


Figure 2.1 Chromatographic purification of untagged Syt1. A. Dot blot using anti-Syt1 antibody and representative fractions from each column are shown. Fraction 12 of HA chromatography was not subjected to Mono S purification simply because I did not know that there were lots of Syt1 in it at that time. **B.** Western blot of HA fractions. The HA column was able to separate Syt1 monomer (fractions 11, 12), dimer (fraction 18, arrow), and degraded Syt1 (fraction 14).

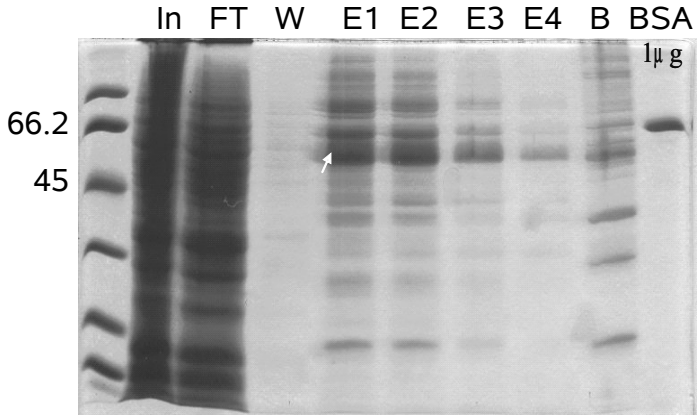


Figure 2.2 Purification of $(His)_6$-Syt1. Coomassie blue-stained gel is shown. After affinity purification, the quality of Syt1 (arrow) was poor, and the concentration was low, too. In: input; FT: flow-through; W: wash; E: eluate; B: beads left; BSA: bovine serum albumin.

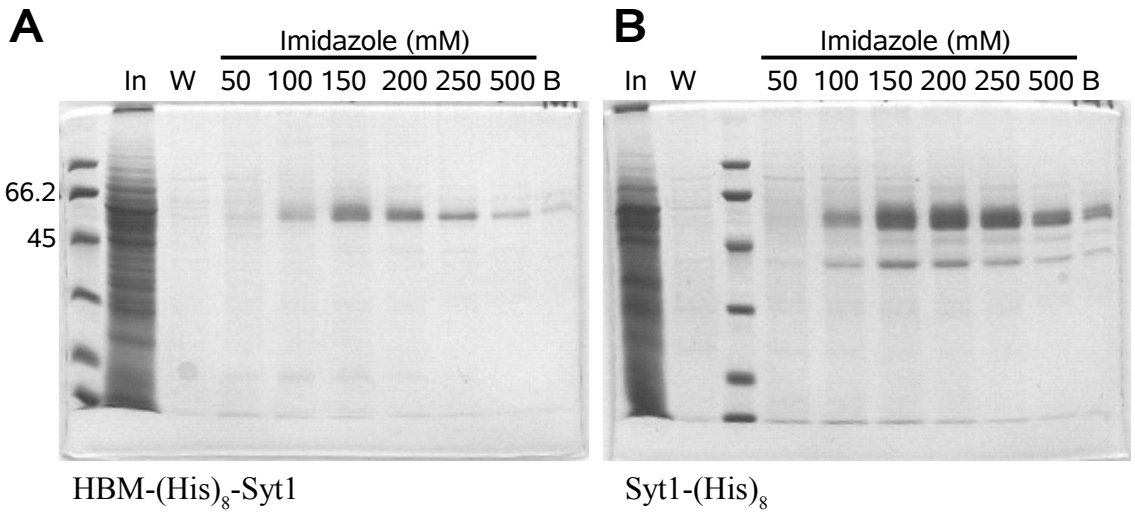
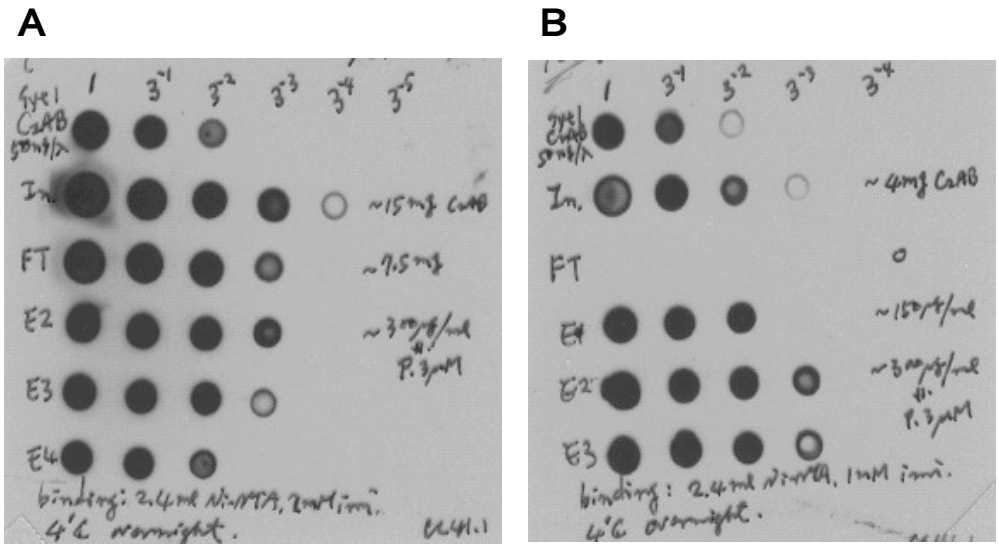


Figure 2.3 Purification of HBM-$(His)_8$-Syt1 and Syt1-$(His)_8$. Coomassie blue-stained gels are shown. **A**. HBM-$(His)_8$-Syt1 and **B.** Syt1-$(His)_8$. The quality and quantity of both expression constructs were significantly increased. In: input; W: wash; B: beads left.



HBM-$(His)_8$-Syt1 Syt1-$(His)_8$

Figure 2.4 Estimation of HBM-$(His)_8$-Syt1 and Syt1-$(His)_8$ concentration. Dot blot with serial diluted samples are shown. Syt1 C2AB domain with known concentration was used as a standard. **A**. HBM-$(His)_8$-Syt1 and **B.** Syt1-$(His)_8$. In: input; FT: flow-through; E: eluate.

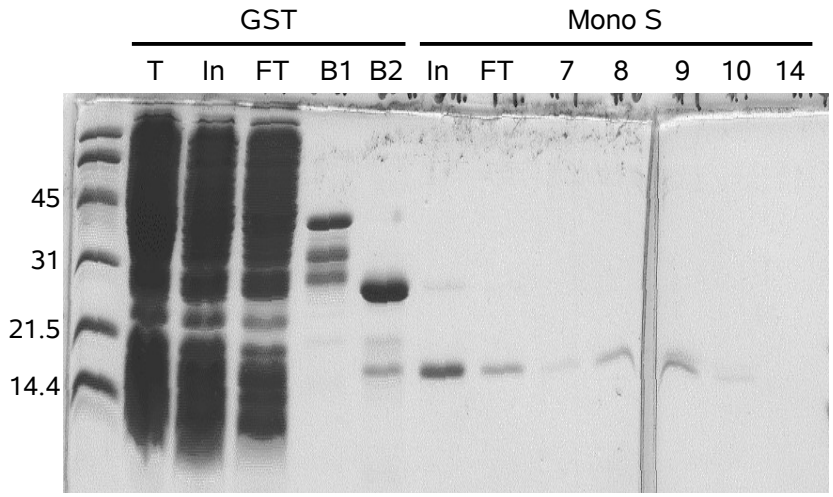


Figure 2.5 Purification of Syb2. Coomassie blue-stained gels are shown. After affinity and ion exchange chromatographic purification, the quality of the purified Syb2 was satisfactory. T: total protein; In: input; FT: flow through; B1: beads before thrombin cleavage; B2: beads after thrombin cleavage. Numbers: Mono S fraction number.

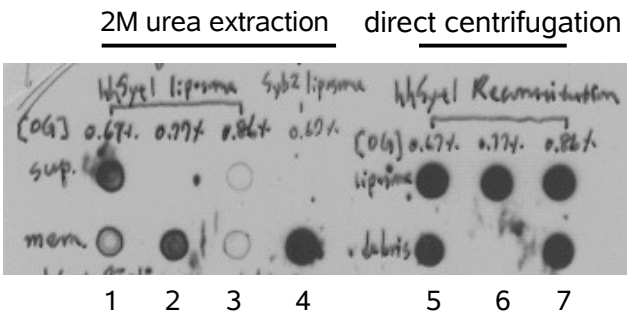


Figure 2.6 Reconstitution of Syt1 at different OG concentrations. OG concentration of 0.67% was used for Syb2 proteoliposome preparation. However, it is clear that Syt1 tends to form aggregation at this concentration (column 5), and the best OG concentration for Syt1 reconstitution is 0.77% (column 6, if total lipid concentration is 5 mM). To check if the "reconstituted" proteins really get integrated into liposomes, 2 M urea was used to remove proteins sticking on the liposomes. Again, reconstitution condition for Syb2 (column 4) is not suitable for reconstituting Syt1 (column 1), and OG concentration of 0.77% is still the best for Syt1 reconstitution (column 2). This OG concentration was selected for subsequent Syt1/Syb2 reconstitution. sup: supernatant; mem: membrane fraction.

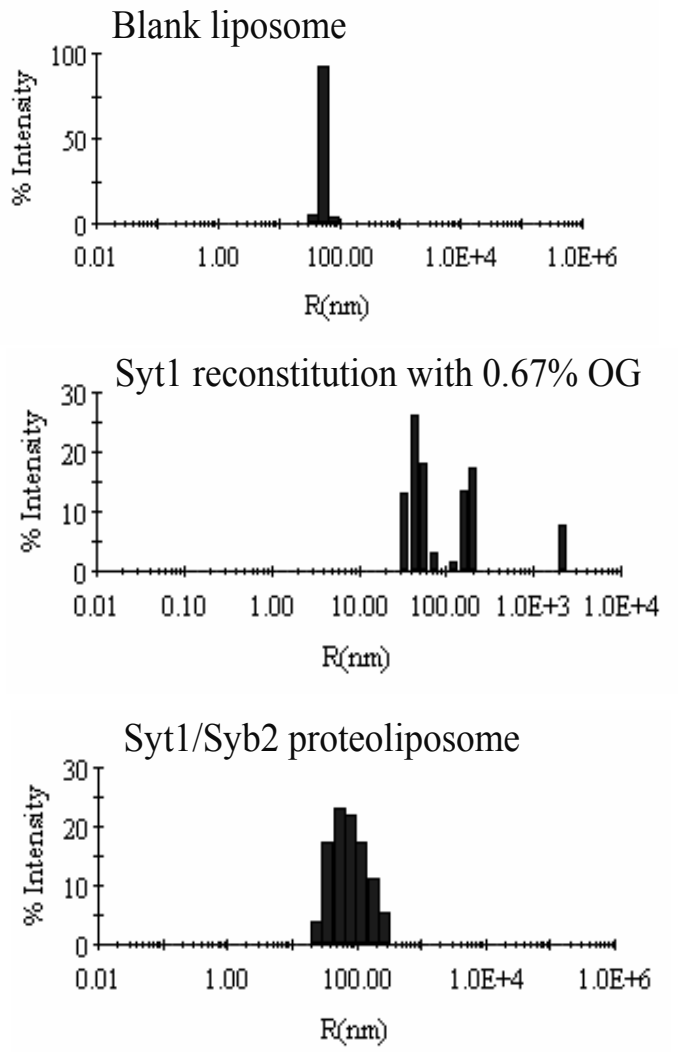


Figure 2.7 Size distribution of reconstituted liposomes. Dynamic light scattering was used for estimating the size distribution of liposomes. Distribution of LUVs made by extrusion method was very homogeneous, and the diameter was about 100 nm (upper panel). When Syt1 was reconstituted using 0.67% OG (previously established OG concentration for reconstituting Syb2, middle panel), the size distribution was very heterogeneous, and there seemed to be particles with large diameter. Using OG concentration of 0.77%, the size distribution of Syt1-containing proteoliposomes was improved tremendously (lower panel),

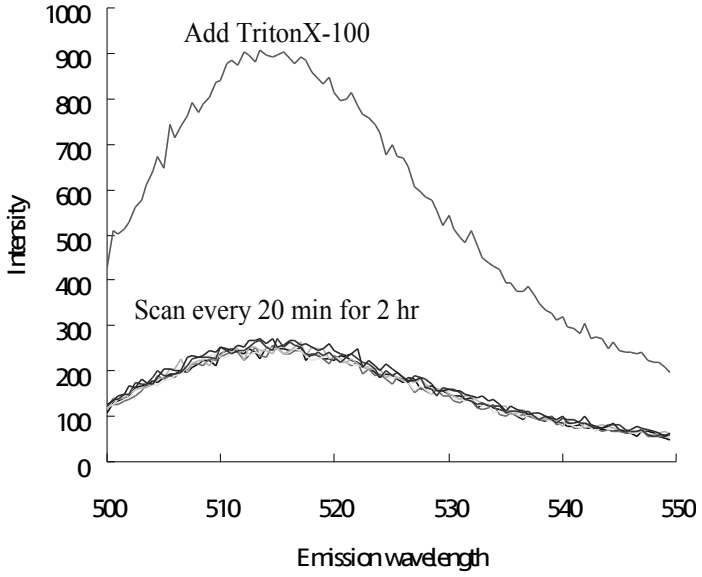


Figure 2.8 Leakage assay of Syt1-containing liposomes. Proteoliposomes loaded with ~100 mM carboxyfluorescein were diluted 80X in buffer, and the leakage of the content (i.e. carboxyfluorescein) was measured by means of dequenching of leaked carboxyfluorescein in a fluorescence spectrophotometer. At the end of measurement, liposomes were lysed by adding 1% Triton X-100 to the cuvette to get maximal fluorescence. No significant sign of leakage was detected within a two-hour period.

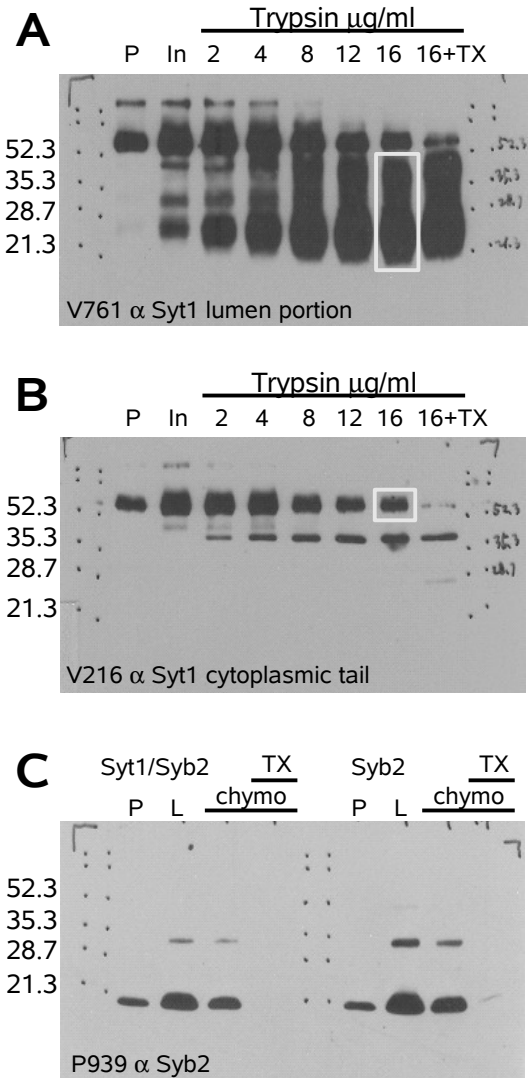


Figure 2.9 Orientation tests of Syt1/Syb2 liposomes. The orientations of incorporated proteins were tested by enzymatic digestion followed by western blot with antibodies against cytoplasmic or lumen portion of the proteins. **A.** With antibody against Syt1 lumen portion (V761), the smear on western film indicated correct-orientated Syt1 (boxed region). **B.** antibody against Syt1 cytoplasmic portion (V216) was employed. Signals remained on the film indicated inversely-incorporated Syt1 (boxed region). **C.** The orientations of Syb2 in Syt1/Syb2 (prepared in 0.77% OG) and Syb2-only (prepared in 0.67% OG) proteoliposomes were also tested. P: pellet; In: input; TX: Triton X-100; L: Syb2 liposome only; chymo: chymotrypsin.

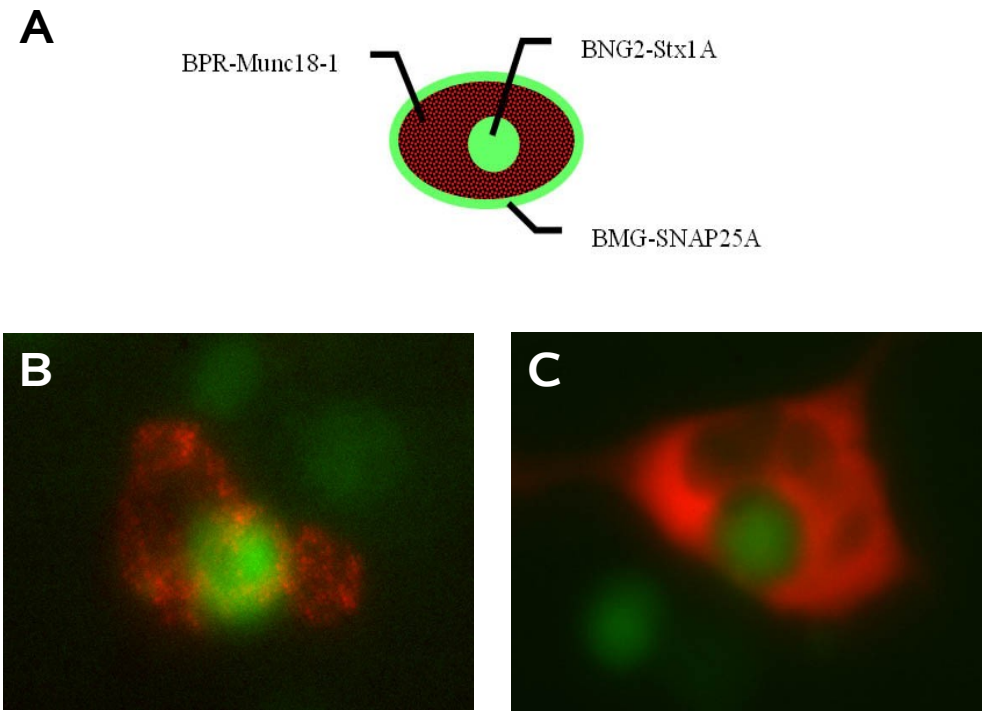


Figure 2.10 The triple-marker transient expression system. A. Ideally, cells co-expressing Stx1A, SNAP25A, and Munc18-1 have green nucleus, green plasma membrane, and red cytoplasm. **B&C.** Epifluorescence images of cells co-transfected with all three expression vectors are shown. The major problem was that the membrane-targeted EGFP could not been seen clearly under epifluorescence microscope.

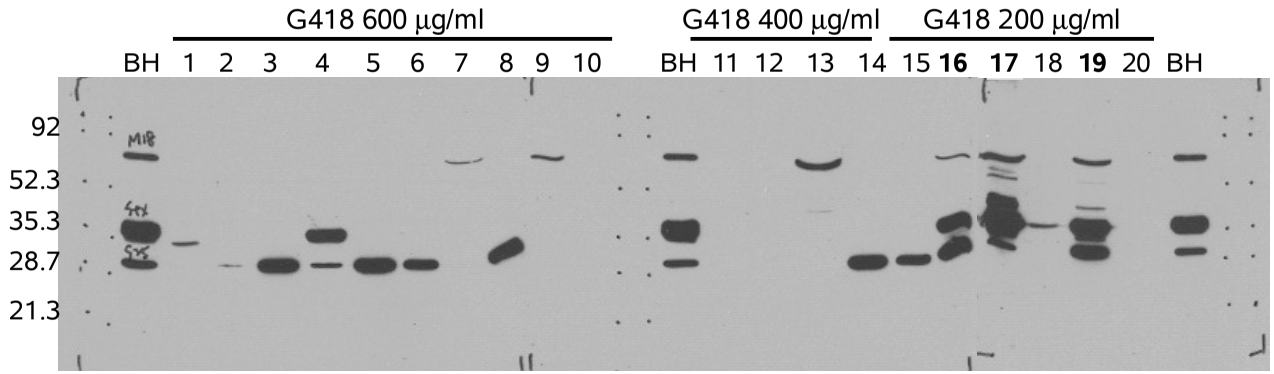


Figure 2.11 Stable cell lines co-expressing Stx1A, SNAP25A, and Munc18-1. HEK293 cells were selected under G418 of 600, 400, and 200 μg/ml. Single colonies of cells were subcloned, and western blot with antibodies against Stx1A, SNAP25A, and Munc18-1 was employed to check the expression pattern. As shown in the figures, clones 16, 17, and 19 contain all three components. BH: rat brain homogenate.

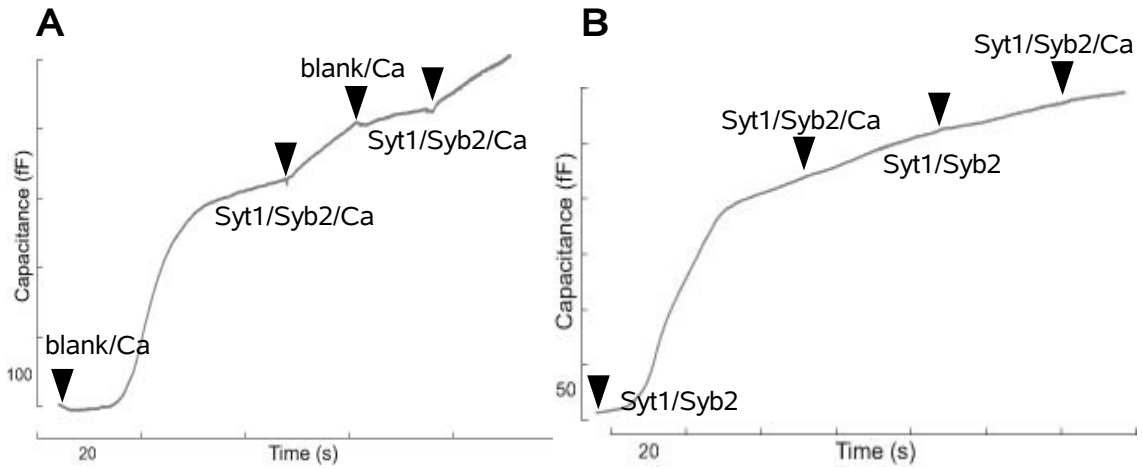


Figure 2.12 Perfusion of artificial liposomes to excised patches. A. After saturating the pipette with blank liposomes/Ca, it appeared that the speed of capacitance increment became faster after moving the patch to Syt1/Syb2 proteoliposomes in the presence of Ca. **B.** The pipette was saturated with proteoliposomes without Ca first, and then move to proteoliposomes with Ca to see if faster capacitance increment (like in **A**) could be observed. Unfortunately, there was nothing happening when Syt1/Syb2 liposomes were applied with Ca, indicating that the result in **A** could be an artifact. Arrowheads: starting time points of perfusion.

Chapter 3. Fusion of endogenous vesicles in excised patches

This chapter is basically the paper entitled “Ca-dependent non-secretory vesicle fusion in a secretory cell”, co-authored by only my mentor Dr. Hilgemann and I. Some more details about the methods are described after the paper, and the derivation of the equations is discussed in detail in Chapter 4.

Ca-dependent non-secretory vesicle fusion in a secretory cell

Tzu-Ming Wang and Donald W. Hilgemann

Department of Physiology, University of Texas Southwestern Medical Center at Dallas, Dallas, Texas 75390, U.S.A.

Running Title: Characterization of non-secretory vesicle fusion

Key words: mast cell, exocytosis, wound repair, amperometry, software lock-in amplifier

Correspondence: Donald Hilgemann

Department of Physiology - ND13.124

UTSouthwestern Medical Center

6001 Forest Park

Dallas, TX 75390-9040

email: donald.hilgemann@utsouthwestern.edu

tel: 1-214-645-6031

fax: 1-214-645-6049

3.1 Abstract

We have compared Ca^{2+}-dependent exocytosis in excised giant membrane patches and in whole-cell patch clamp with emphasis on the rat secretory cell line, RBL. Stable patches of 2-4 pF are easily excised from RBL cells after partially disrupting actin cytoskeleton with latrunculin A. Membrane fusion is triggered by switching the patch to a cytoplasmic solution containing 100-200 μM free Ca^{2+}. Capacitance and amperometric recording show that large secretory granules (SGs) containing serotonin are mostly lost from patches. Small vesicles that are retained (non-SGs) do not release serotonin, or other substances detected by amperometry, although their fusion is reduced by tetanus toxin (TeTx) light chain. Non-SG fusion is unaffected by *N*-ethylmaleimide, phosphatidylinositol-4,5-bis-phosphate ($PI(4,5)P_2$) ligands, such as neomycin, a PI-transfer protein that can remove PI from membranes, the PI(3)-kinase inhibitor, LY294002, and $PI(4,5)P_2$, PI(3)P and PI(4)P antibodies. In patch recordings, but not whole-cell recordings, fusion can be strongly reduced by ATP removal and by the nonspecific PI-kinase inhibitors, wortmannin and adenosine. In whole-cell recording, non-SG fusion is strongly reduced by osmotically-induced cell swelling, and subsequent recovery after shrinkage is then inhibited by wortmannin. Thus, membrane stretch that occurs during patch formation may be a major cause of differences between excised patch and whole-cell fusion responses. Regarding Ca^{2+} sensors for non-SG fusion, fusion remains robust in synaptotagmin (Syt) VII -/- mouse embryonic fibroblasts (MEFs), as well as in PLCδ1, PLC δ1/δ4, and PLCγ1 -/- MEFs. Thus, Syt VII and several PLCs are not required. Furthermore, the Ca^{2+} dependence of non-SG fusion reflects a lower Ca^{2+}

affinity (K_D ~71 μM) than expected for these C2-domain-containing proteins. In summary, we find that non-SG membrane fusion behaves and is regulated substantially differently from SG fusion, and we have identified an ATP-dependent process that restores non-SG fusion capability after it is perturbed by membrane stretch and dilation.

3.2 Introduction

Mast cells of hematopoietic origin play a central role in inflammatory responses by releasing numerous substances that modulate immune responses (Metcalfe et al., 1997). Fusion of large secretory granules (SGs) to the plasma membrane underlies degranulation of mast cells, and much experimental effort has focused on the regulation of this exocytotic process (Burgoyne and Morgan, 2003; Sagi-Eisenberg, 2007). In addition to exocytosis of SGs, another type of fusion that does not result in clear step-wise changes of membrane capacitance (non-SG) was reported in rat peritoneal mast cells twenty years ago (Almers and Neher, 1987). While the step-size of non-SG events is not readily resolved by capacitance recording, the increase of cell capacitance mediated by such fusion events can be of very large magnitude (Almers and Neher, 1987). The sources of membrane involved in non-SG fusion as well as the underlying fusion mechanisms remain rather enigmatic. Data from chromaffin cells indicate that non-SG fusion requires relatively high Ca^{2+} (~100 μM) and is ATP-dependent, but neurotoxin-insensitive. Non-SGs in chromaffin cells are not likely to represent acetylcholine-containing synaptic-like microvesicles (Xu et al., 1998), and it is striking that non-SG fusion can also be massive in CHO and 3T3 cells (Coorssen et al., 1996), as well as BHK and HEK293 cells (Yaradanakul et al., 2008), and as described in some detail in this article, in RBL, MEF, and INS-1 cells. The prevalence of large-scale Ca^{2+}-activated non-SG fusion processes in both secretory and non-secretory cell lines suggests that the non-SG fusion may be important for cell survival. Since the non-SG pool can exceed 50% of the total surface membrane area and the requirements for cytoplasmic Ca^{2+} are rather high (Yaradanakul et

al., 2008), it seems reasonable that this membrane pool is involved in wound repair of the plasma membrane.

Present understanding of membrane fusion relies strongly on studies of transmitter/hormone release from neurons (Bronk et al., 2007; Sakaba et al., 2005) and endocrine cells (Burgoyne and Morgan, 2003), and the homotypic fusion of yeast vacuoles (Ostrowicz et al., 2008). In these cases, the SNARE (soluble *N*-ethylmaleimide-sensitive factor attachment protein receptor) proteins are clearly implicated to initiate fusion, and in general are thought to do so by associating and perturbing the two membranes involved. As introduced in an accompanying article (Yaradanakul et al., 2008), phosphatidylinositides and their derivatives are presently thought to importantly modify SNARE-dependent fusion processes (De Matteis and Godi, 2004). Evidence from PC12 cells suggested that formation of phosphatidylinositol 4,5-bisphosphate ($PI(4,5)P_2$) microdomains at syntaxin clusters can activate the exocytotic sites (Aoyagi et al., 2005). $PI(4,5)P_2$ appears to be required for 'priming' of vesicles in pancreatic β-cells (Olsen et al., 2003) and the yeast vacuole fusion process (Mayer et al., 2000), and it regulates the releasable vesicle pool size in chromaffin cells (Milosevic et al., 2005). In dense core vesicle fusion, $PI(4,5)P_2$ acts via the calcium-dependent activator protein for secretion (CAPS) to increase the initial rate of fusion (Grishanin et al., 2004; Loyet et al., 1998). Furthermore, $PI(4,5)P_2$ is suggested in a liposome-liposome fusion system to promote fusion directly via its interaction with the neuronal Ca^{2+} sensor, Synaptotagmin (Syt) I (Bai et al., 2004). Finally, phospholipase Cs (PLCs), which cleave $PI(4,5)P_2$ to produce diacylglycerol (DAG) and inositol triphosphate (IP_3), are implicated to regulate some

vesicle fusion processes (Fukami et al., 2001; Jun et al., 2004) and, in general, the conversion of large phospholipid head groups to smaller ones is expected to favor fusion of phospholipid vesicles. Other phosphoinositides, such as phosphatidylinositol 3,4,5-triphosphate (PIP_3) and phosphatidylinositol 3-phosphate (PI(3)P) also play important roles in membrane trafficking, including events leading up to fusion (Lindmo and Stenmark, 2006). For example, inhibition of a class IA PI(3) kinase (PI(3)K), which produces PIP_3, reduces receptor-mediated degranulation in mast cells (Ali et al., 2004). PI(3)K-C2α, which produces mainly PI(3)P, is also required for the ATP-dependent priming of SGs in neurosecretory cells (Meunier et al., 2005).

From the various methods used to monitor membrane fusion, membrane capacitance measurements give the highest signal and temporal resolution, and the development of improved excised patch models to analyze and manipulate fusion would have important experimental advantages of high resolution recording with free access to the cytoplasmic membrane side. Efforts to date have used chromaffin cells (Dernick et al., 2003) and the insulin-secreting INS-1 cells (MacDonald et al., 2005). In the present study, we describe the use of giant excised membrane patches (pipette diameter ~10-15 μm, patch size ~2-4 pF) to attempt to preserve more fusion-capable vesicles in a configuration that allows free access to the cytoplasmic side. In short, we have found that partial disruption of the actin cytoskeleton facilitates the formation of stable patches from some cell types and preserves more fusion-capable vesicles in patches from all cell types tested. Nevertheless, our experience is that SGs are easily lost from excised patches, presumably because they are not docked in a stable fashion at the plasma membrane in RBL cells. The non-SG

fusion is much more robust, and we describe here several fundamental characteristics of this type of fusion, including its dependence on ATP and Ca^{2+}. The results define clear differences between non-SG and SG fusion processes and provide new insights into the physical basis of non-SG fusion.

3.3 Methods

Cell culture

Adherent RBL-2H3 cells were cultured in Dulbecco's Modification of Eagle's Medium (DMEM, Mediatech, Herndon, VA) supplemented with 15% fetal bovine serum (FBS). Cells were plated on uncoated Petri dishes 1-2 days before experiments and collected by treating them with 0.25% trypsin/EDTA solution. Serotonin and 5-hydroxytryptophan (0.2 mM/each) were also added to cells for amperometric recording one day before the experiments to increase the formation of SGs (Mahmoud and Fewtrell, 2001; Williams et al., 1999). After treatment with trypsin, the cells were resuspended in the culture medium and left in the CO_2 incubator for 30 min. Cells were then treated with Latrunculin A (50-100 ng/ml) at 37°C for 5 min before experiments, as this treatment clearly facilitated the formation of giant excised patches with fusion-competent vesicles. Mouse embryonic fibroblasts (MEFs) were cultured in DMEM supplemented with 10% FBS and penicillin-streptomycin. They were plated on cell culture dish 1-2 days before experiment and collected as stated above. The Syt VII-deficient MEF cell line was provided by Dr. Thomas Südhof (UTSouthwestern), the PLCδ1- and PLCδ1/δ4-deficient MEF cell lines were provided by Dr. Kiyoko Fukami (Tokyo University), and the PLCγ1-deficient cell line was provided by Dr. Graham Carpenter (Vanderbilt University)

Solutions

Unless otherwise stated, the patches were excised into a solution containing in mM; 140 NaCl, 1 $MgCl_2$, 0.3 EGTA, 20 HEPES, pH 7.3. ATP and GTP were added at final

concentration of 2.4 and 0.3, respectively. Membrane fusion was triggered by the same solution with 0.5 $CaCl_2$ (i.e. 0.2 mM free Ca^{2+}), usually without ATP and GTP. For whole-cell recording with RBL cells, both the cytoplasmic and extracellular solutions contained in mM; 40 NaCl, 90 *N*-methyl-D-glucamine (NMG), 1 $MgCl_2$, 0.01 EGTA, 10 HEPES, pH 7.3 adjusted with MES. Relatively large-diameter pipette tips (4-6 μm i.d.) were employed in whole-cell recording to allow fast exchange of the cytoplasm via pipette perfusion. Using this low conductance solution, the cell time constants (30-60 μs) were large enough to use square wave perturbation for capacitance measurements, as described subsequently. A solution with 0.2 mM free Ca^{2+}, highly buffered with nitrilotriacetic acid, was infused into the cell to induce membrane fusion. The complete composition was in mM; 15 NaCl, 90 NMG, 3 $MgCl_2$, 5 $CaCl_2$, 10 nitrilotriacetic acid, 10 HEPES, pH7.3 (adjusted with MES). All free Ca^{2+} values given in this article were calculated with WEBMAXC (http://www.stanford.edu/~cpatton/maxc.html) (Patton et al., 2004). Other solutions employed were 'FVPP solution' (Huang et al., 1998), a phosphatase inhibitor cocktail (110 NaCl, 5 NaF, 0.1 Na_3VO_4, 2 EDTA, 10 $Na_4P_2O_7$, 20 HEPES), 'EDTA buffer solution' (140 NaCl, 2 EDTA, 20 HEPES), and 'protein dialysis solution' (140 NaCl, 0.3 EGTA, 0.3 $ZnSO_4$, 20 HEPES, 2 β-mercaptoethanol). For whole-cell experiments presented in Fig. 3.7, the 'standard solutions' described previously (Yarandanakul et al, 2008) were employed. For Fig. 3.8, results in panel A employed the 'standard solution' with 70 mM NMG substituted for LiOH. The cytoplasmic solution was modified by addition of 200 mM sucrose and dilution by 30% to generate the hyper- and hypoosmotic solutions, respectively. In panel B, the 'standard solutions' were employed with NMG

(aspartate) reduced by 80 mM to generate the hypoosmotic extracellular solution. The 'control solution' was generated by adding 160 mM sucrose to this solution. In panel C, the solution given above for RBL cells was employed. Hyperosmotic cytoplasmic solution was generated by addition of 200 mM sucrose, and hypoosmotic extracellular solution was generated by deletion of 80 mM NMG.

Recording software

Capacitance measurement software, Capmeter 6, was developed in MATLAB using its data acquisition toolbox (R2006b; The MathWorks, Natick, MA). Three major on-line functions were developed: 1. a software lock-in amplifier, 2. routines for continuous cell parameter determination via square wave voltage perturbation, and 3. data smoothening and deglitching routines. To synchronize the timing of analog output and input, a 1 mV trigger signal was added to the sine or square wave, usually at 100 Hz. For the lock-in amplifier function, the phase-sensitive detector of the program multiplied the current with either an in-phase or an orthogonal reference signal. The direct-current (DC) component of the product was extracted by averaging to cancel the non-DC noise. The double of the DC component was assigned to X or Y where the in-phase or the orthogonal reference signal was employed, respectively. The optimal phase angle, θ, was determined by small changes of the optimally adjusted capacitance compensation of the patch clamp as follows:

$$\theta = \theta_0 - \arctan\left(\frac{X - X_0}{Y - Y_0}\right) \qquad (3.1)$$

where θ_0 (and X_0, Y_0) and θ (and X, Y) represent values before and after changing

capacitance compensation, respectively.

For square wave perturbation (time-domain method; see Fig. 3.1), continuous square pulses were applied, and current transients were recorded and analyzed *on-line* using Capmeter 6. The average current (i.e the DC component) was first calculated and subtracted from the total current. The resulting trace was then divided into two parts and fitted separately to exponential functions. For curve-fitting, the steady-state current (I_∞ = '*b*' in Fig. 3.1A) was determined as the asymptote of current from the averages of three consecutive data sections of equal length (*A*, *B* and *C*; Fig. 3.1B dashed sections):

$$b = \frac{B^2 - AC}{2B - C - A} \qquad (3.2).$$

This equation is the solution for the asymptote, *b*, of the three simultaneous functions, $A = b + Y^* e^{-t/\tau}$, $B = b + Y^* e^{-(t+x)/\tau}$, and $C = b + Y^* e^{-(t+2x)/\tau}$. The steady-state current was then subtracted from the trace, and the data range from the peak current to a point located at ~3τ (estimated as peak current times e^{-3}; Fig. 3.1B solid section) was used for fitting via linear regression to determine the slope and intercept at zero time, namely -1/τ and *ln(a-b)*, respectively.

Our routine to calculate cell capacitance (*Cm*), membrane resistance (*Rm*), and access resistance (*Ra*) can be derived as follows. Membrane voltage ($V_{(t)}$) approaches a steady state (*Vss*),

$$Vss = \frac{VcRm}{Ra + Rm} \qquad (3.3),$$

as a function of the command voltage, *Vc* (peak-to-peak step = *2Vc*), with a time

constant, τ, of *Cm/(1/Ra + 1/Rm)*. For square wave perturbation, membrane voltage during the pulse is given by the exponential function,

$$V_{(t)} = -fVss + Vss(1+f)(1-e^{-t/\tau}) \qquad (3.4),$$

where *f* is the fraction of *Vss* across the membrane at the end of the voltage step of duration, *Δ*. Solving for *f* with *t=Δ*,

$$f = \frac{1-e^{-\Delta/\tau}}{1+e^{-\Delta/\tau}} \qquad (3.5),$$

and membrane voltage at the beginning of the voltage step is

$$V_{(0)} = \frac{-fVcRm}{Ra+Rm} \qquad (3.6).$$

From the steady state current,

$$b = \frac{Vc}{Ra+Rm} \qquad (3.7),$$

and the peak current,

$$a = \frac{Vc - V_{(0)}}{Ra} \qquad (3.8),$$

the solutions for *Ra, Rm* and *Cm* are

$$Ra = \frac{Vc(1+f)}{a+bf} \qquad (3.9),$$

$$Rm = \frac{Vc(a-b)}{b(a+fb)} \qquad (3.10),$$

$$Cm = \tau\left(\frac{1}{Ra} + \frac{1}{Rm}\right) \qquad (3.11).$$

Our algorithm was verified by using it to retrieve cell parameters from model cell simulations using the MATLAB component, Simulink, as well as our own routines. In the absence of noise and a filter function, the algorithm retrieved simulated cell parameters with errors of ~1 ppm. With cell parameters that would be considered experimentally unacceptable (e.g. 200 pF, a Ra of 20 MΩ, a Rm of 50 MΩ, and voltage oscillation at 200 Hz), the algorithm still retrieved the parameters with an accuracy of 99.9%.

Signals were usually acquired at 100 kHz, and digital filtering was performed by averaging signals in an adjustable time window. Data were usually digitized at 100 Hz and a running mean/median filter was applied to the digitized data when data smoothening/deglitching was desired. Program Capmeter 1 was used with the hardware lock-in amplifier, serving as a plain data recorder with digital filtering and data smoothening/deglitching functions. The programs are available for download at http://capmeter.googlepages.com.

Patch clamp and data acquisition

We used National Instruments (Austin, TX) board PCI-6052E to generate the command potential and collect signals, and we used an Axopatch-1D (Molecular Devices, Sunnyvale, CA) for patch clamp. Electrode tips were dipped in molten hard dental wax (Kerr Corporation, Romulus, MI) before cutting and polishing to reduce stray capacitance. For excised patches, electrodes with ~15 μm inner diameters were employed. The giant patch was ‘excised’ by essentially aspirating the cell into a second

pipette with a sharp, unpolished edge (Hilgemann and Lu, 1998). The patches were positioned in front of a temperature controlled (~30°C) solution outlet immediately after excision. Membrane fusion was triggered by moving the patch to a solution outlet containing 0.2 mM free Ca^{2+}. Capacitance and conductance were measured using the Lindau-Neher method (Lindau and Neher, 1988). Sine waves generated by Capmeter 6 with 20 mV peak-to-peak amplitude at 2 kHz were applied to the cell. The current output from the patch clamp was low-pass filtered at 10 kHz. When sine wave perturbation was employed, the optimal phase angle was determined as described above. When patch amperometry was employed, a hardware lock-in amplifier (SR830; Stanford Research Systems, Sunnyvale, CA) was employed, as it allowed a higher signal-to-noise ratio at oscillation frequencies >3 kHz. Sine waves with V_{rms} of 20 mV at 10 kHz were usually employed. The signals were recorded by Capmeter 1.

For whole-cell recording, with ~5 μm inner diameter pipette tips, membrane fusion was initiated via perfusion of Ca^{2+}-containing (nitrilotriacetic acid-bufferd) solution through a quartz capillary with a 40 μm outlet, manipulated within the patch pipette to a distance of 50-100 μm from the cell opening (Hilgemann and Lu, 1998). Square wave 20 mV (peak-to-peak) perturbation at 0.5 kHz was employed in all experiments presented in this article for whole-cell capacitance recording, with cell parameters determined by Capmeter 6 as described above.

Patch amperometry

The setup was connected according to Dernick et al. (Dernick et al., 2005) with some

modifications. In brief, two Axopatch-1D amplifiers were used. One of the headstages was connected to the bath for capacitance recording, the other one was connected to the carbon electrode for recording the amperometric current, and the patch pipette was the ground. The carbon electrodes were made from 7 μm carbon fibers (C005711; Goodfellow corporation, Oakdale, PA) and quartz capillaries (Polymicro Technologies, Phoenix, AZ). Flowable silicone windshield/glass sealer (Permatex, Hartford, CT) was used to insulate the carbon fiber, and the tip was cut to expose the carbon surface before installing (Fig. 3.2A). The carbon electrode was installed through the infusion line of the pipette holder and connected to the amplifier using 3 M KCl and Ag/AgCl wire. The electrode was moved toward the patch as close as possible and a holding potential of 0.7 V was applied. The amperometric signals were low-pass filtered at 5 kHz by the amplifier and digitally filtered again by averaging signals acquired in a time period of 1 ms using Capmeter 1 and then digitized at 500 Hz.

Recombinant proteins and antibodies

The wild-type and mutant (E234Q) GST-tetanus toxin light chain fusion constructs were kindly provided by Dr. Thomas C. Südhof (UTSouthwestern, Dallas). Recombinant proteins were purified from bacteria, BL21, according to the manufacture's protocol (GE Healthcare, Piscataway, NJ). Proteins were eluted from the beads using reduced glutathione and then dialyzed against buffer containing $ZnSO_4$ (final concentration, 200 nM). Anti-PI(4,5)P_2 antibody (1:50) was kindly provided by Dr. Kiyoko Fukami (The University of Tokyo, Tokyo). Anti-PI(3)P (1:100) and anti-PI(4)P (1:100) antibodies were purchased from Echelon Biosciences (Salt Lake City, UT). The PI-transfer protein (140

μg/ml) was generously provided by Dr. Vytas A. Bankaitis (The Univeristy of North Carolina, Chapel Hill).

Data analysis

Except for experiments done with MEFs, all experiments were performed in a one control vs. one test result pattern. For excised patch records, the capacitance traces were well described by a mono-exponential function with a small linear drift component:

$$Y = a + b(1 - e^{-kt}) + ct \qquad (3.12)$$

where b represents the theoretical maximal amplitude, k is the rate constant used in statistical analysis, and c represents the slow component. For patches with $b<0$, the amplitude of zero was assigned. To calculate the ratio of active patches, we used a threshold of 50 fF to define active and inactive ($b<50$ fF) patches. We mention that we were not able to determine the capacitance of excised patches routinely, so that normalization of results to patch size is impossible. Also, we mention that methodologically-induced capacitance changes during solution changes are in the range of a few tens of femtofarad (see Fig. 3.3A). Thus, we typically calculated a ratio of active patches for a group of experiments, as well as the average capacitance changes, and rate constants were collected and compared from the patches that met the 'active' criterion.

For whole-cell experiments, phase-sensitive detection was also used off-line to improve the signal-to-noise performance of the exponential fitting routine with square wave perturbation, as follow. The phase angle was determined at which C, the absolute capacitance determined by the exponential analysis, and Y had the highest cross-

correlation. Offline phase angle adjustment was done using equations,

$$X_{adj} = X\cos(\theta) + Y\sin(\theta) \qquad (3.13)$$

$$Y_{adj} = -X\sin(\theta) + Y\cos(\theta) \qquad (3.14)$$

The whole-cell capacitance traces were fitted with a delayed-mono-exponential function:

$$C = a + b(1 - e^{-k_1 t})^n \cdot (1 - e^{-k_2 t}) + ct \qquad (3.15)$$

where a and b represent initial cell capacitance and vesicle pool size, respectively. The constants k_1 and n reproduce reasonably the observed delays, and the constant k_2 is the rate constant used for statistical comparison. The Ca^{2+}-activated conductance increase was used as a reference for determining the t_0 point.

For all bar graphs of 'amplitude' and 'ratio of active patches' panels, numbers in the bars give the total number of patches; for other panels, numbers represent the numbers of valid data after removing outliers with Grubbs' test. Statistical significance was determined by Student's t-test. All error bars in figures represent S.E.M.

3.4 Results

Distinct vesicle populations in RBL cells

To study serotonin secretion in RBL cells, carbon electrodes were prepared as described in Methods and mounted in a quartz tube with a Ag/AgCl electrode that could be inserted into the patch pipette (Fig. 3.2A) to detect the released serotonin. In the cell-attached configuration, application of 2 μM calcium ionophore, A23187, induced massive membrane fusion that was detected as an increase of membrane capacitance (Fig. 3.2B). Two types of vesicle populations were observed; one is the secretory granule (SG) pool accompanied with big capacitance steps and amperometric spikes; the other one observed here at the end of the trace contains vesicles of much smaller size that are evidently not filled with serotonin (non-SG pool). The capacitance steps of SG fusion were in the range of tens of fF, indicating that the diameters of the SGs were in the submicro- to micrometer range, consistent with values reported by most (Spudich and Braunstein, 1995) but not all investigators (Smith et al., 2003).

Empirically, we found that treating the cells with latrunculin A facilitated both the formation of giant excised patches and the preservation of non-SGs on the patches. Nevertheless, our success rate to preserve SGs in the RBL giant patches was prohibitive for routine studies, perhaps because the SGs are not predocked at the membrane in RBL cells (Smith et al., 2003) in a stable fashion and/or because the docking is disrupted by membrane suction. We also attempted to develop the INS-1 cell line and bovine chromaffin cells for excised giant patch studies. Our experiences with the INS-1 cells were similar to those reported for RBL cells. Batch-to-batch variability was even greater,

and we were not able to identify a reliable line. Bovine chromaffin cells readily allowed seal formation with large-diameter pipettes, but excised giant patches were not stable with significant solution flow, thereby greatly limiting their use. In short, the giant patch approaches did not facilitate, in our hands, excised patch analysis of SG fusion processes. Occasionally, high resolution recordings were indeed obtained with clear capacitance steps and amperometric spikes as shown in Figs. 3.2C, D for RBL patches. The fact that only non-SGs were preserved on the great majority of excised patches implies that the non-SGs are in close vicinity to the plasma membrane and might be associated with the membrane physically.

Non-SG fusion is SNARE-dependent but NEM-insensitive in excised patches

The physical characteristics of non-SGs are not well established, and we therefore used the giant patch approach to analyze this process in some detail, in particular to manipulate the cytoplasmic membrane side. Capacitance traces were fitted as described in Methods (*Y* in Fig. 3.3B), defining patches with capacitance increases smaller than 50 fF as inactive (see Fig. 3.3A, left panel), and patches with more robust non-SG fusion as active patches (see Fig. 3.3A, right panel). Notably, the exocytotic response often appeared to be followed by an endocytotic response, even in the presence of Ca^{2+}, when ATP and GTP were present on the cytoplasmic side (see Fig. 3.3A)

To test whether the non-SG fusion is SNARE-dependent, we treated the patches with tetanus toxin (TeTx) light chain at a concentration of 200 nM for 2min. As shown in Fig. 3.3C, both the amplitude and the ratio of active RBL patches were decreased significantly

by treatment with the wild-type toxin, compared with the inactive mutant form, indicating that the fusion is indeed SNARE-dependent. Notably, however, the treatment of patches with 1 mM *N*-ethylmaleimide (NEM) did not block non-SG fusion (Table 3.1), implying that SNARE cycling, which is blocked by NEM (Xu et al., 1999), is not required for non-SG fusion in excised patches. As mentioned in the Introduction, it is reported that non-SG fusion in bovine chromaffin cells is toxin-insensitive (Xu et al., 1998). A simple explanation for the discrepancy to our data is that RBL and chromaffin cells use different sets of SNAREs for non-SG fusion, and the SNAREs accounting for non-SG fusion in chromaffin cells are toxin-resistant. Another possibility is that the SNAREs of non-SGs in chromaffin cells are complexed, such that neurotoxins cannot cleave them (Chen et al., 2001). As described later, in whole-cell recordings with RBL cells the amplitudes of non-SG fusion usually exceed 50% of the basal cell capacitance (Figs. 3.6 and 3.7). Non-SG fusion in chromaffin whole-cell recordings is usually substantially smaller in absolute and relative terms (Xu et al., 1998), indicating that its non-SG pool is much smaller and possibly already primed and therefore toxin-resistant.

ATP hydrolyzing processes support non-SG fusion

It is well documented that ATP is required by the NSF (NEM-sensitive factor) to disassemble the cis-SNARE complex in the SNARE cycle (Jahn et al., 2003; Whiteheart et al., 1994), and that the addition of Vam7p (a soluble SNARE) bypasses the requirement of ATP in the yeast vacuole fusion system (Thorngren et al., 2004). Since non-SG fusion in excised patches is NEM-insensitive (Table 3.1), one might expect that the non-SG fusion would be ATP-independent. In whole-cell recording, replacement of ATP with a

non-hydrolyzable analogue was found to significantly reduce non-SG fusion (Yarandanakul et al., 2008), and we describe here that ATP-hydrolyzing processes clearly support non-SG fusion in excised patches. In some cases, the absence of ATP caused complete failure of fusion, and the ability to fuse was restored when ATP and GTP were added back to the solution (see Fig. 3.4A). That ATP is the critical factor for restoration of fusion was verified in several experiments in which ATP was applied without GTP (Figs. 3.3A, 3.4B, and data not shown). Further experiments supported the notion that ATP hydrolysis is essential to maintain fusion because the non-hydrolyzable ATP analogue, AMP-PNP (2 mM), did not substitute for ATP (Fig. 3.4B).

Since the decrease of fusion after removal of ATP, and with substitution by AMP-PNP, takes much longer than expected for washout of ATP from the patch, ATP-hydrolyzing reactions clearly are not part of the final fusion process. The ATP mechanism(s) that support fusion in the longer-term could involve the phosphorylation of target proteins/lipids and/or the use of ATP in other types of energy-dependent reactions, as in the case of NSF. In favor of the former idea, we could preserve the fusion capability in the absence of ATP by adding EDTA to the solution (see Table 3.1). Since magnesium is the only divalent ion in our recording solution, we hypothesized that ATP is used to phosphorylate one or more targets, and that activity of the counteracting phosphatases would be magnesium-dependent.

ATP-dependent generation of $PI(4,5)P_2$ is not critical

As mentioned earlier, $PI(4,5)P_2$ has been implicated in multiple aspects of fusion processes, and we therefore tested if the role of ATP is to maintain $PI(4,5)P_2$ in the

excised patches and, as well, if the cleavage of $PI(4,5)P_2$ is required for triggering membrane fusion. Application of a very high concentration of neomycin (500 μM) to bind $PI(4,5)P_2$ (Eberhard et al., 1990), and probably other anionic phospholipids, in a magnesium-free solution did not block non-SG fusion. Furthermore, application of anti-$PI(4,5)P_2$ antibodies in FVPP solution did not block fusion (Table 3.1), although the concentrations of antibody employed potently blocked $PI(4,5)P_2$-senstive currents in excised patches (e.g. outward Na/Ca exchange current, not shown). These results suggest that neither the synthesis nor the hydrolysis of $PI(4,5)P_2$ can be a requirement for non-SG fusion. As shown in Table 3.1, no protecting effect of these agents was observed when they were applied in a magnesium-containing solution. Clearly, magnesium-dependent hydrolysis of $PI(4,5)P_2$ cannot be the mechanism of run-down of the fusion process.

Non-SG fusion is probably phosphatydialinositide-independent

Since the ATP mechanism does not appear to involve $PI(4,5)P_2$, we used other inhibitors to probe potential phosphorylation targets. As shown in Table 3.1, treating the patches with staurosporine, a broad-spectrum protein kinase inhibitor, was not able to block fusion. Interestingly, treating the patches with high concentrations of wortmannin (4 μM) and adenosine (0.5 mM), which inhibit multiple classes of PI(3)Ks and PI(4)Ks (Balla et al., 2008), significantly decreased non-SG fusion (Fig. 3.5), implying that PI(3)P and/or PI(4)P might be responsible for the ATP dependency. Therefore, we tested for roles of individual lipids by applying specific antibodies in whole-cell recording experiments before triggering fusion via Ca^{2+} infusion. Impressively, neither anti-PI(3)P nor anti-PI(4)P antibody was able to block or slow down non-SG fusion, and a

combination of both antibodies was also ineffective (Fig. 3.6). It seems unlikely that PIP_3 is involved because another PI(3)K inhibitor, LY294002, also did not block fusion in excised patches (Table 3.1). In fact, pretreatment of the cells with a submicromolar concentration of wortmannin, which blocks PIP_3 producing PI(3)K, and treatment with genistein, an inhibitor of tyrosine kinases that are typically activated in this pathway (Galetic et al., 1999), failed to block non-SG fusion in whole-cell recording (data not shown).

We also used a PI transfer protein (Mousley et al., 2007) to remove PI from excised patches to test if there is any role of PIs at all in non-SG fusion. As shown in Table 3.1, fusion was not inhibited by a concentration that we found to inhibit the ATP-dependent stimulation of Na/Ca exchange current in excised patches, a process determined to reflect phosphorylation of PI (Nasuhoglu et al., 2002). All of these negative results support the conclusion that non-SG fusion is PI-independent. The actual targets and the specificity of wortmannin and adenosine in this study must therefore be questioned, and as described next there is an important difference between excised patches and whole-cell responses in this regard.

Wortmannin/adenosine-insensitivity of non-SG fusion in whole-cell recording

The pronounced inhibition of non-SG fusion in excised patches by the wortmannin/adenosine combination was unexpected, as we had tested these agents in whole-cell recording of Ca^{2+}-induced fusion in BHK fibroblasts and found no effect. Therefore, we reexamined this issue in whole-cell recordings from both RBL cells and

from BHK cells, as shown in Fig. 3.7. Individual records from RBL cells are shown in Fig. 3.7A with the composite statistics for the data set. Pipette perfusion of solution containing 200 μM free Ca^{2+}, as in Fig. 3.6, was initiated 2 min after opening cells. In control cells, the average increase of cell capacitance was 119±12%, and it occurred with a time constant of 26±3.1 s (n=5). In cells that were perfused with 5 μM wortmannin and 0.5 mM adenosine for 2 min, the average increase was 114±5.2%, and the time constant was 30±3.1 s. Fig. 3.7B shows the equivalent results for whole-cell BHK recording using NCX1 to initiate membrane fusion. In these experiments, we tested adenosine (0.5 mM) alone, as well as the same adenosine/wortmannin combination used in RBL cells. After 2 min cytoplasmic infusion of the respective solutions, the peak exchange currents were modestly reduced (~ 20%), at just the level of significance, but the percent-increment of cell capacitance upon activating exchange currents (~65% in these experiments) was unchanged by either treatment.

Non-SG fusion in whole-cell recording is blocked by cell swelling

The lack of a significant effect of these treatments in whole-cell recordings suggested that a mechanism becomes important to maintain fusion capability in the excised patch, which is not critical under the usual conditions of whole-cell recording. We reasoned that mechanical forces, exerted on the membrane during seal formation and patch excision, might disrupt the fusion machinery, and ATP-hydrolyzing processes would then become essential to restore the fusion capability. In other words, membrane stretch and/or distention (i.e. flattening of invaginations) might be an important factor, and accordingly we tested whether cell swelling might mimic effects of seal formation and excision. Fig.

3.8 describes two sets of results from BHK cells and one from RBL cells, all demonstrating strong inhibition of non-SG fusion by cell swelling. We note that different BHK batches were employed in the two data sets in A and B, and that, as often was the case, the average capacitance responses were substantially different in the different batches.

In Fig. 3.8A, cell swelling was induced in BHK cells by employing a cytoplasmic solution with addition of 200 mM sucrose, and shrinkage was induced by employing a cytoplasmic solution diluted by 30%. Exchange currents were activated 2 min after opening cells and cell shape changes had clearly occurred. As shown in the upper bar graphs, the peak exchange currents were not significantly affected. Cell swelling was associated with a 76% decrease of the fusion response ($p<0.01$), whereas cell shrinkage with hypoosmotic cytoplasmic solution was without effect.

Next, we tested whether cell swelling by hypoosmotic extracellular solution also reduces membrane fusion. As shown in Fig. 3.8B, placement of cells in extracellular solution with NMG reduced by 80 mM (‘hypoosmotic outside’) for 2 min prior to activating exchange current caused an 85% decrease of membrane fusion. To examine whether the effect of swelling is reversible, we placed cells for 2 min in hypoosmotic solution and then moved them back into isoosmotic solution for 2 min prior to activating exchange currents. During the protocol, swelling and shrinkage of cells was clearly visible. As shown for ‘post-swelling’, the fusion response was partially restored. In a fourth experimental group, we tested whether the restoration of fusion might be blocked by wortmannin. Using the same protocol to allow restoration of fusion, inclusion of 5 μM

wortmannin in the pipette solution significantly decreased the recovery of fusion responses after swelling ($p<0.05$). From 3 observations, we did not find the adenosine/wortmannin combination to be more effective (data not shown). Overall, these whole-cell results support the notion that membrane perturbation and/or stretch strongly inhibits non-SG fusion, and that biochemical processes are required to restore fusion capability.

Fig. 3.8C presents the equivalent experiments for RBL cells with membrane fusion induced by pipette perfusion of cytoplasmic solution with 200 μM free Ca^{2+}, as in Figs. 3.6 and 3.7. Results for hyperosmotic cytoplasmic solution (with 200 mM added sucrose) and for hypoosmotic extracellular solution (with NMG reduced by 80 mM) are very similar to results with BHK cells. Membrane fusion responses are reduced by 74 and 75%, respectively. Thus, the high sensitivity of the non-SG fusion process to inhibition by cell swelling is verified across two cell lines and with different protocols to induce membrane fusion.

Synaptotagmin VII and PLCs are not required for non-SG fusion

The non-SG pool in RBL cells can almost double the total surface membrane area (Figs. 3.7A, 3.8C), and it seems likely that this pool will become involved in the wound repair of the plasma membrane. Lysosomes have been suggested to be the major vesicles that undergo Ca^{2+}-dependent exocytosis in nonsecretory cells (Jaiswal et al., 2002) and in wound repair (Chakrabarti et al., 2003). Furthermore, it is suggested that the Ca^{2+} sensor in lysosomal fusion is the synaptotagmin VII (Syt VII) (Chakrabarti et al., 2003). To test if Syt VII is important for non-SG fusion, we examined membrane fusion in mouse

embryonic fibroblasts (MEFs) because, like RBL cells, they also have robust non-SG fusion, and knockout lines are available. As shown in Fig. 3.9B, non-SG fusion was still robust in *syt7* knockout MEFs (6 observations). Unlike results with RBL cells, it was notable that non-SG fusion in MEFs often involved fusion of large vesicles (Fig. 3.9, arrowheads). Whether such large vesicles can be lysosomes is unclear, but it is evident that both types of fusion are still robust when *syt7* is ablated. We conclude that Syt VII cannot be an important Ca^{2+}-sensor for non-SG fusion.

Relevant to the potential importance of phosphoinosidites, phospholipases, and alternative possible Ca^{2+} sensors, we tested whether this type of membrane fusion in MEF cells was affected by deletion of three PLCs. No evident differences were found for membrane fusion episodes recorded using MEFs and excised patches from MEFs with deletion of PLCγ1, PLCδ1, and both PLCδ1 and PLCδ4 versus a control MEF cell lines (n>3 for all observations, data not shown).

Ca^{2+}-dependence of non-SG fusion in excised RBL patches

To characterize further the Ca^{2+}-sensing machinery of non-SG fusion, we triggered fusion with different concentrations of free Ca^{2+} in excised patches from RBL cells. The concentration-response data for free Ca^{2+} versus rate of fusion shows a Hill coefficient of ~2, an apparent K_D of ~71 μM, and a maximal rate constant of ~2.1 s^{-1} (Fig. 3.10). We point out that the two data points at low Ca^{2+} concentrations were weighted to force the fit through these points. Without weighting, these points were not well described by the fit, and the Hill coefficient was ~3. We stress, however, that Hill coefficients determined for non-SG fusion in BHK cells were also in the range of 2 (Yaradanakul et al., 2008). It is

reported that the Hill coefficient and $[Ca^{2+}]_{1/2}$ of Syt VII is ~3.6 and ~0.9 μM, respectively, in a liposome binding assay (Wang et al., 2005). Therefore, these results support further a conclusion that Syt VII is not likely to be the sensor for non-SG fusion.

3.5 Discussion

In this article, we have described some efforts over several years to maintain and to monitor Ca^{2+}-dependent exocytosis in giant excised patches. While non-secretory vesicle fusion could be routinely monitored, in our hands neurotransmitter release was seldom maintained in the excised patches. Even in the case of non-SG fusion, we found it necessary to pretreat cells with agents that disrupt cytoskeleton to maintain fusion. That membrane stretch and distention may readily disrupt fusion has been verified in whole-cell recording with both BHK and RBL cells. We discuss first the methodological and experimental problems encountered and then the data sets on non-SG fusion.

Membrane fusion in excised giant membrane patches

In principle, the giant patch methods should facilitate multiple types of fusion studies. Excised patches allow free access to the cytoplasmic side for a wide range of possible manipulations by exogenous factors, including proteins. Nevertheless, after several years we have not been able to establish conditions that allow routine recordings of neurotransmitter release in excised patches, including efforts with several cell types. For example, bovine chromaffin cells readily allowed seal formation with large-diameter pipettes, but excised patches are not stable with significant solution flow, thereby greatly limiting their use. Overall, our experience is that the ability to induce neurotransmitter release is very easily lost or destroyed during excision procedures, and we did not overcome this limitation for routine work. Given the strong inhibition of fusion by cell swelling, we strongly suspect that mechanical factors are of most importance, but it also remains possible that important soluble factors are lost from the patches.

Non-secretory fusion in excised patches

In this article, we have described that non-SG membrane fusion is both robust and massive in several cell lines when studied by whole-cell voltage clamp, while responses in excised patches showed a large degree of variability. Specifically, we found that the ratio of active patches varied substantially from batch to batch of the cells, as well as with the length of time after isolation. Sometimes, no fusion at all was observed in excised patches from an entire batch of cells, although whole-cell responses were robust and highly reliable in the same cell batch. Also, individual cell batches were encountered in which fusion did not run down in the absence of ATP and EDTA (data not shown), while the loss of fusion over time in ATP-free solution was highly reliable in other batches. For these reasons, all experiments were done in a one control vs. one test result pattern, as pointed out in Methods and results can only be compared with data collected at the same time using the same batch of cells.

Our overall interpretation is that in our routine whole-cell configuration all available vesicles eventually fuse to the plasma membrane, as there are no inactive cells, and the percentage increase of cell area so impressively large in relation to the number of docked vesicles observed (Yaradanakul et al., 2008). Since the fusion process is strongly inactivated by cell swelling, it seems likely that membrane stretch, or flattening of invaginations, disrupts the fusion machinery with restoration requiring ATP-dependent processes of an unknown nature. Remodeling of cytoskeleton is an attractive but unproved possibility. Our results on run-down in patches (Fig. 3.4 and Table 3.1) are consistent with ATP being used to phosphorylate one or more targets that maintain the

fusion capability, whereby dephosphorylation evidently occurs by a magnesium-dependent phosphatase. In this regard, it is known that the enzymatic activities of tyrosine phosphatases vary with culture conditions and ages (Cool and Blum, 1993; Pallen and Tong, 1991). It would not be surprising therefore that the activities of the kinases, as well as other phosphatases, also vary among batches of cells and the length of time after isolation. Unless AMP-PNP is added to the pipette solution, the ATP-dependency of non-SG fusion in whole-cell recordings is usually not significant (Yaradanakul et al., 2008). Thus, the excised patch model naturally produces many more sources of variability that do not exist in the intact cells. This is at once an advantage for identifying important partial mechanisms of fusion, and its regulation, and a disadvantage because the reproducibility of experiments is significantly decreased.

ATP-sensitivity of non-SG fusion

Results from this article further support our conclusion that neither $PI(4,5)P_2$ nor its metabolism modulate significantly non-SG fusion, when the trigger Ca^{2+} concentration is high (Yaradanakul et al., 2008). Depletion of $PI(4,5)P_2$ at the plasmalemma does not block non-SG fusion in M1 receptor-expressing BHK cells (Yaradanakul et al., 2008), and we have shown here that multiple $PI(4,5)P_2$ ligands, namely neomycin and $PI(4,5)P_2$ antibodies, do not affect Ca^{2+}-induced fusion in the excised patches from RBL cells. While the antibodies employed might not have sufficient affinity to block high-affinity functions of phosphoinositides in fusion, we expect that the high concentration of neomycin employed (500 μM) would bind non-selectively all phosphoinositides and thereby inhibit their functional role. Only one positive result concerning

phosphoinositides was obtained, namely that non-SG fusion was blocked by the combined application of wortmannin and adenosine. However, both reagents may have non-specific effects. At high concentrations, wortmannin inhibits mitogen-activated protein kinase (MAPK) (Ferby et al., 1996) and myosin light chain kinases (MLCK) (Nakanishi et al., 1992). Since non-SG fusion is not blocked by staurosporine (Table 3.1), although staurosporine blocks the activities of MLCK, MAPK might be an interesting target to be tested in future work.

In conclusion, it does not seem surprising that non-SG membrane fusion, which is probably used for cell wound repair, is regulated in substantially different fashion from the release of neurotransmitters, and the ATP dependence of non-SG fusion likely represents a unique regulatory mechanism that comes into play after mechanical perturbation of the fusion system.

Wound repair and non-SG fusion

While we infer that the membrane fusion process examined in this study is related to the membrane wound response, the membrane compartment(s) involved in non-SG fusion are still enigmatic. In addition to lysosomes, a novel organelle, named the enlargosome, is proposed to mediate membrane repair (Borgonovo et al., 2002). Fusion of the enlargosome in rat PC12 cells is TeTx-insensitive, which is different from non-SG fusion in RBL cells as described here. However, it has also been reported that the wound repair machinery is TeTx-sensitive in other model systems (Togo et al., 1999), and as mentioned earlier, non-SG fusion is toxin-insensitive in bovine chromaffin cells (Xu et al., 1998). One evidence supporting the notion of wound repair by non-SG fusion comes

from the low Ca^{2+} sensitivity of the Ca^{2+}-sensor, which is relevant to the ongoing debate about the role of Syt VII in membrane repair (McNeil and Kirchhausen, 2005). Syt VII is reported to be a high affinity Ca^{2+} sensor in SGs of PC12 cells (Wang et al., 2005). The low Ca^{2+} sensitivity of non-SG fusion could in principle ensure that these vesicles are not affected by normal Ca^{2+} signaling in the cell, and that they would fuse with the plasmalemma only when bulk Ca^{2+} influx comes from the wounded sites.

In addition to establishing a new approach to study non-SG fusion with giant excised patches, we have developed computer software that can be useful to other groups to implement both time-domain and frequency-domain methods (Lindau and Neher, 1988). In our experience, the time-domain method is especially useful when clamp time constants are relatively long (e.g. hundreds of microseconds in cardiac myocytes). With these approaches, we have delineated several new characteristics of non-SG fusion. First, while this type of fusion is SNARE-dependent, it is not NEM-sensitive in excised patches. Second, while this type of fusion is ATP-dependent, it is not phosphoinositide sensitive. Thus, the ATP-sensitivity comes about by a novel mechanism that might prove to be relevant to other types of membrane fusion. Third, the non-SG vesicles are presumably pre-docked at the plasmalemma to remain attached to excised patches, and in this regard more detailed ultrastructural analysis will be of paramount importance for further progress. Fourth, the Ca^{2+}-dependence of this mechanism is unlikely to represent the function of the putative Ca^{2+} sensor, Syt VII. And finally, the non-SG fusion mechanism strongly inhibited by cell swelling and, presumably, membrane stretch. Together, these results have established multiple, potentially novel directions for future

studies of non-SG fusion, which is likely to be an important partial reaction of the ubiquitous membrane wound repair response.

Acknowledgements. We thank Vincenzo Lariccia for assistance and discussions, Marc Llaguno and Alp Yarandanakul for advice and criticism. We thank Dr. Thomas Südhof (UTSouthwestern), Dr. Kiyoko Fukami (Tokyo University), and Dr. Graham Carpenter (Vanderbilt University) for generously providing MEF cell lines, Chengchen Shen, Mei-Jung Lin for assistance, and Dr. Vladislav Markin (UTSouthwestern) for discussions of mathematical methods.

3.6 Some more methodological details

Isolation of rodent peritoneal mast cells

The protocol for isolating mouse peritoneal mast cells is modified from the standard protocol (Mundroff and Wightman, 2002) and the one on the Protocol Online website (http://www.protocol-online.org). The mast cell buffer used in the protocol contains (in mM) 140 NaCl, 5 KCl, 10 glucose, 2 $CaCl_2$, and 1 $MgCl_2$, pH7.4. One mouse is used for each preparation, and the mouse is anesthetized by inhalation of a mixture of CO_2 and O_2.

Clean the abdomen of the anesthetized mouse with 70% ethanol, and then inject about 3 ml of the mast cell buffer and 2 ml of air into the peritoneal cavity using a syringe with 25-G needle. It is important to inject some air into the peritoneal cavity as it increases the yield in the following shaking step. After injection, retract the needle from the mouse, and the injected buffer and air is retained inside the cavity. Shake the mouse 10-15 times and also massage its abdomen for about 2 min. Shaking the mouse harshly or for too many times may result in contaminating mast cells with excessive amount of red blood cells.

Use a forcep to hold the skin and cut a little hole with sharp scissors. Do not make the incision too large because the buffer containing mast cells may flow out. Collect buffer in the peritoneal cavity with a plastic (or glass) pipette, and pour the cell into a 15-ml centrifuge tube. Wash the cavity several times with a total amount of about 15 ml buffer, and collect the buffer in the same tube. Centrifuge the tube at 200 g for 5 min, discard the supernatant, and then resuspend the cell pellet in 8 ml RBL culture medium

(DMEM/15% FBS). Plate 2 ml of cells on each 35-mm uncoated Petri dish, and culture the cells at 37°C in a humidified CO_2 incubator. The diameter of mast cells is about 10 μm. The reason for using a Petri dish rather than a cell culture dish is to prevent attachment of cells onto the dish, thus reducing the mechanical force needed for isolating the cells. In my experience, too much mechanical force triggers fusion of SGs prior to experiments, which reduces the releasable pool as a consequence.

For isolating rat peritoneal mast cells, I use a total amount of about 40 ml mast cell buffer, and follow the same protocol as described above. Rat peritoneal mast cells are bigger than those from mouse.

Tips for reducing noise in patch amperometric recordings

The recording chamber and solution lines are filled up with electrolytes, which behave like a big antenna and collect a lot of electrical noise. In regular patch clamp configuration, the ground is connected to the recording chamber, thus, electrical noise from outside of the system is sequestered. As mentioned in section 3.3, for patch amperometric recordings, the ground has to be in the pipette, and the headstage for recording capacitance etc. is connected to the bath. In this case, noise introduced by the recording chamber and solution lines is not neutralized, and will be detected by the headstage used for capacitance recording. Here, I will describe some tips that were found to be useful according to my experience.

1. Keep solution lines as short as possible

   As mentioned above, the solution lines are like an antenna network. The longer

the lines, the stronger the signal. Don't get me wrong, the signal received by the lines is electrical noise. So you have to keep the lines as short as possible. To do so, you will need to move the solution reservoirs (e.g. syringes) close to the recording chamber. In addition, also keep the volume of solution in the reservoirs as small as possible to further reduce the noise (I use <0.5 ml for each reservoir, 3 reservoirs total). Theoretically, reducing the chamber size also reduces the noise, however, it is not practical. The solution level in the chamber is increasing during the experiment because of the continuous solution flow. If the chamber is too small, solution might flow out of the chamber during the experiment, and the speed of increasing stray capacitance caused by the increasing solution level will be too fast, which might saturate the lock-in amplifier during the recording.

2. Use negative pressure, but not solution switch to control the flow

   In regular configuration, the solution switch is connected between the reservoir and the line to control the solution flow. It seems natural and reasonable to connect the system this way, however, disasters happen 99% of the time if this configuration is used for patch amperometry. The inevitable scenario is that when you turn the switch after everything is ready, the potential difference (presumably) between the reservoir and the chamber blasts your precious seal. It does not help to use other types of solution stopper (e.g. metal clamp) as long as they are placed between the reservoir and the chamber. The only way I found useful is to apply negative pressure to hold the solution, and release it to start the solution flow. This special reservoir is made by two 3 ml syringes. One of the syringes is cut at ~0.5

ml scale mark, and the other one is cut very close to the tip. The tip portions of the syringes are glued together using a hot melt glue gun. One end of the reservoir is connected to the solution line, and the other end is connected to a solution switch. This switch is used to hold the air pressure, and when it's turned on, the pressure is released.

3. Shield everything

   Since the solution lines behave like an antenna network, it is import to shield the exposed lines as much as possible. You may make a Faraday cage around the perfusion system using aluminum foil or plate, and make sure that the Faraday cage is grounded. If the cage is not grounded, it might introduce even more noise into the system.

   In addition to shielding the solution lines, you may want to shield the temperature control system as well. Although the circulating water in the temperature control system is not electrically connected to the recording chamber, it may still carry detectable noise into the recording system. Use aluminum foil to wrap all the tubings, and again, make sure that the foil is properly grounded.

   There is still something more to be shielded. Tubings for hydraulic micromanipulator (if used), the headstage for recording amperometric current, and the carbon electrode (except the portion that is inserted into the pipette holder) also need to be shielded. It is to reduce the noise in amperometric current.

4. Ground yourself

Human body is conductive, and receives lots of electrical noise from the environment, too. If you put your hands into the Faraday cage (e.g. turning on the solution flow or moving the patch) without grounding yourself, huge noise signals are introduced into the system, and sometimes may even bast the seal. The best way to solve this problem is to ground yourself electrically by keeping one of your hands on the Faraday cage during the experiment. If you ground yourself to the cage right before moving your hands into it, charges discharged from your body may still result in spike-like artifacts in the record.

5. Do not let anything shaking in the Faraday cage

   Anything shaking in the cage may also give spike-like or low frequency noise in the record. Possible objects include tubings of hydraulic micromanipulator, suction line, carbon electrode, and aluminum foil etc. Also avoid strong air flow around the cage to minimize potential source of vibration.

6. Do not move your chair during recordings

   I know it sounds crazy. If I move my chair during the recording, not always, but sometimes it does give some artifacts in the record. So, I recommend you not to move anything (including your body) during the recording.

Besides tips mentioned above, you can increase the oscillation frequency and amplitude of the sine waves to reduce noise in capacitance records. And as a last resort, you can apply running mean/median filter to suppress the noise. However, keep in mind that the kinetics of the trace is slower than original after filtering.

Tips for making carbon electrodes

The standard ways for making carbon electrodes are well documented (Chow and von Rüden Ludolf, 1995; Dernick et al., 2005; Mundroff and Wightman, 2002), and the electrodes are even commercially available. However, these types of electrodes do not fit my patch clamp headstage, and it is also difficult for me to make a delicate and specialized pipette holder like that. So, I decided to design one that is easy to make, and easy to use (Fig. 3.2A). Before describing the procedures for making electrodes, I would like to thank Dr. Marc Llaguno, whose specialty is carbon nano-tube. He is the one who suggested me to insulate the electrodes using windshield glass sealer. Steps for making carbon electrodes are listed below.

1. Glue two quartz tubings together

   Cut a ~2 cm piece of 75 μm (i.d.) quartz tubing, and a ~8 cm piece of 200 μm (i.d.) quartz tubing. Insert about half of the smaller one into the large one, and then glue them together using a hot-melt glue gun. Do not use too much glue. You may disperse just a tiny amount of glue evenly by spinning the tubing at the hot metal opening of the glue gun for a while, and then pull the tubing away quickly from it.

2. Insert carbon fiber into the quartz tubing

   This is the most difficult part, and you will need to make some tools for manipulating the fibers first. The first tool is used to separate one single carbon fiber from a bundle of fibers. I found the best tool is the quartz tubing. You can use two old carbon electrodes (made by quartz tubing, too) for this purpose. The

second tool is for inserting the carbon fiber. Cut a ~0.5 cm piece of PE tubing, and then stick one of the sharp ends of a forcep into it. The protruding PE tubing is used for inserting the fiber.

To isolate a single carbon fiber, put a small bundle of fibers (~4 cm long) on a white paper, and then separate a single fiber using two quartz tubings under a stereomicroscope. After a single fiber is isolated, hold the quartz electrode (prepared in step 1) in one of your hands, and hold the forcep in the other hand. Press the protruding PE tubing of the forcep on the carbon fiber to hold it, and then move the quartz tubing toward the tip of the fiber with an angle of ~30 degree. You may find it difficult insert the fiber because of some electrostatic force. If it's difficult to do so on the paper, try to do it on aluminum foil. It sometimes helps. Keep moving the quartz tubing to insert the fiber, and hold the fiber still with the other hand until the fiber no longer gets inserted or reaches desired length. If you can not get carbon fiber inserted long enough into the quartz tubing, try to “sweep” the fiber into the tubing using the forcep made previously. Be careful not to break the fiber. I usually keep ~0.5 cm fiber handing outside of the quartz tubing. If the tip portion is too long, the recorded amperometric current might be quite noisy. If it is too short, you may not reuse the electrode since the tip is cut every time before experiment to expose fresh carbon surface (step 4).

3. Insulate the protruding carbon fiber

Put some flowable silicone sealer on a pipette tip (I use yellow tip), hold the yellow tip, and then adjust the focus of the stereomicroscope so that you can see

the ball-shaped sealer clearly. Under the microscope, immerse the protruding carbon fiber into the ball-shaped sealer and let it stand for few seconds. The sealer goes into the quartz tubing owing to capillary effect. Pull the electrode out of the sealer slowly in a direction that is perpendicular to the electrode. The carbon fiber is bent in this way, but nothing bad is going to happen because it is quite flexible. The boundary between the ball-shaped sealer and the fiber moves (retracts) slowly from the quartz tubing to the tip of the carbon fiber. If the boundary retracts too fast, the coating may be too thick, and there will be many silicone droplets dispersing along the carbon fiber. Similar phenomenon may happen when the sealer is too sticky. In this case, you may dilute the sealer with some 100% ethanol. It usually happens when the sealer is opened and stored for a period of time.

Cure the sealer at 50°C overnight, or at room temperature for at least one day before using. The thickness of the insulation is usually ~1 μm (estimated by eyes). If it is too thick, you may need to improve the coating step (move even slower), or dilute the sealer a little bit.

4. Expose carbon surface and back-fill the electrode with 3 M KCl

Put the carbon electrode on a clean paper under the stereomicroscope, and adjust the magnification as high as feasible. Cut the carbon electrode with a scalpel blade under the microscope. Be sure not to cut at a silicone droplet (if any) because you may not be able to move the carbon electrode close enough to the

membrane with an enlarged silicone shield around the tip.

To fill the electrode with KCl, you need you make an adaptor first. Cut a ~10 cm quartz tubing (i.d. 75 μm), insert it into a 25 G needle, and then glue them together using a melted (with lighter) yellow tip. Connect the adaptor to a 0.45 μm filter and a syringe filled with 3 M KCl, and then insert the adaptor tip into the electrode. Push the syringe to fill the electrode with KCl, and pull the adaptor out of the electrode slowly. Make sure that the carbon fiber is in contact with KCl solution, and also avoid bubbles in the electrode.

5. Insert Ag/AgCl wire and seal the end

Install the carbon electrode to the pipette holder through the infusion line outlet in a direction that is from the holder to the headstage. Insert a 76.2 μm (0.003 inch) Ag/AgCl wire into KCl solution in the electrode, and then seal it with hot-melt glue. To seal the junction with hot-melt glue, push some glue out of the glue gun (form a little “ball”), cool the glue a little bit, and then immerse the junction into the glue ball. Pull the glue gun a little bit away from the electrode in a direction that is perpendicular to the electrode, and then further cool down the glue with some airflow. Pull away the glue gun quickly and cut the “tail” with a razor blade. If the glue ball is too hot when you immerse the junction into it, the seal will be too tight. In this case, you probably won't be able to reuse the Ag/AgCl wire because you may break it when you try to remove it from the electrode. However, if the glue ball is not hot enough, the seal may be too loose, and solution will

evaporate from the electrode and the electrode will disconnect with the wire in a short period time. In addition, also note that for convenience, the Ag/AgCl wire has been soldered to a regular wire that is connected to the headstage.

Efforts for preserving secretory vesicles in excised patches

The original plan for my thesis was to study "regulated" exocytosis in giant excised membrane patches. Sadly, after doing amperometric recording, it turned out that the membrane fusion I was studying for these days are not fusion of large SGs. To accomplish the original goal (without much luck though...), I addressed this issue from two directions. One is to change the way the patch is excised, and the other direction is to use cell lines with more SGs. They are summarized below.

1. Use air bubbles to excise the patch

   The idea came from the literature that people lift the cell in the air for ~1 s to make sure that the patch is in inside-out, but not in vesicle-like configuration (Dernick et al., 2003). Special apparatus was made to control the attacking strength of bubbles. However, technically it was difficult to excise the patch this way, and the method itself did not help to preserve SGs in the patches, either.

2. Use electrical pulses to excise the patch

   To avoid mechanical force that might disrupt the fusion machinery, I also tried to open a hole on the cell using electroporation with a platinum wire. It did not help, either.

3. Inject solution to blow up the cell

In my standard, a good scientist should be able to solve the problem creatively. As a good-scientist-to-be, I decided to use a creative way to excise the patch, and entertain myself at the same time to relieve my stress in these days. Blowing up the cell with a second pipette theoretically can avoid mechanical force applied on the cell during the excision procedure, and the cells were indeed opened in this way and exposed the cytoplasmic side to the solution flow. Surprisingly, SGs were still not preserved in this way.

Under the microscope, detachment of optical-dense materials from the plasma membrane were observed clearly when the solution was injected into the cell. Presumably, the cytoskeleton is included in the optical-dense materials, and the detachment of it may result in pulling off the SGs which are likely to be associated with it.

4. Permeabilize the cell using 40 μM β-escin

The β-escin is a detergent that makes holes on the membrane and allows molecules with a molecular weight of up to 10 kDa to pass through the pores (Fan and Palade, 1998). It has been used to make perforated patches in different cell types like neuron (Sarantopoulos et al., 2004) and myocyte (Dougherty et al., 2008). As a continuous effort to reduce the mechanical force applied on the cell during the excision procedure, I applied 40 μM β-escin from the extracellular side for ~30 s to create pores for Ca^{2+} entry. Using mouse peritoneal mast cells, Ca^{2+} did enter the β-escin-treated mast cells, and triggered massive SG fusion.

However, it is often observed that SGs started to fuse before Ca^{2+} was applied, indicating that the ER and/or other internal Ca^{2+} stores were also permeabilized.

In addition, β-escin has to be prepared freshly, and the potency varies among different preparations. Thus, it is difficult to explain negative results because one can not tell if the cell is not permeabilized enough, or the fusion machinery is really jeopardized under testing condition. For the same reason, it is also difficult to keep the initial Ca^{2+} concentration in the cell the same in different experiments. Because of these reasons, I eventually decided not to continue this approach.

5. Change the direction of excision

   As mentioned previously, some optical-dense materials are detached from the membrane when cells are blew up. It is also true when the standard excision procedure is used. Since the SGs may be associated with these materials, I tried to move/excise the patch in different directions in hope that the dense materials might be preserved better. It is difficult to describe the actual motion in words, and unfortunately I don't have a movie for that, either. The point is to excise the patch gently, and try to keep the dense materials attach to the membrane.

   When RBL cells were used, some SGs were indeed preserved in excised patches using this method, but the success rate was quite low, and there were only few SGs in a patch. It is presumably owing to the low abundance of SGs and the heterogeneity of the RBL line (will be discussed later).

To increase the number of secretory granules, I have tried the followings.

1. Pre-incubate cells with 10 μM ATP

   It is reported that the cytosolic Ca^{2+} modulates the supply of release-competent vesicles in chromaffin cells (Smith et al., 1998), and low basal Ca^{2+} concentration may result in the depriming of the vesicles (Neher, 2006; Smith et al., 1998). To ensure that SGs remain "primed" before the experiment, I incubated cells with 10 μM ATP to elevate basal cytoplasmic Ca^{2+} concentration. ATP is the ligand for purinergic P2-type receptors, which are also expressed in RBL cells (Clifford et al., 1998; Osipchuk and Cahalan, 1992). An ATP concentration of 30 μM had been used to elevate intracellular Ca^{2+} concentration of astrocyte to ~1 μM (Kanemaru et al., 2007). I was using 10 μM ATP incubation for 10 min, and hoping that the Ca^{2+} concentration in RBL could fall into a sub-micromolar range. The outcome was that I still could not preserve SGs in the patches after treating the cells with ATP.

2. Pre-incubate cells with 100 nM PMA

   PMA (phorbol 12-myristate 13-acetate) is a DAG analogue that has been shown to increase the primed SGs in chromaffin cells (Gil et al., 2001). However, the RBL cells became enlarged and transparent after treating them with 100 nM PMA for only a few minutes. In addition, PMA is a potent carcinogen, and will contaminate the recording chamber and might be inhaled by mouth when making the seal. Thus, I decided to stop this approach.

3. Use bovine chromaffin cells

I obtained bovine chromaffin cells from Dr. Joseph P. Albanesi's lab in UTSouthwestern. As mentioned in the Discussion previously, it is easy to get giant excised patches from bovine chromaffin cells. However, the patches were not stable under significant solution flow, making it impractical to use them for other excised patch experiments.

4. Use rat chromaffin cells

   Rat chromaffin cells were isolated according to paper published by Dr. Frederick W. Tse in University of Alberta, Canada (Xu et al., 2005), and his supervision. Unlike bovine chromaffin cells, chromaffin cells from rat are quite small. I did not have any success with them.

5. Use mouse peritoneal mast cells

   Mouse peritoneal mast cells can be prepared easily, and they have tons of SGs. However, they are quite small, and difficult to patch. The success rate for me to excise a giant patch from them is zero. So, I stopped this approach.

6. Use rat peritoneal mast cells

   The rat peritoneal mast cells are substantially bigger than the mouse ones. But unlike mast cells from mouse, their shape is flat. It is possible to make small excised patches from them, but for giant patches, it is not quite easy. I did not continue this approach both because of the low success rate, and because knockout rat does not exist.

7. Subclone RBL cells

After extensive amperometric recordings in cell-attached configuration, I found out that the RBL cell line I was using is quite heterogeneous, and does not contain many SGs. To overcome this problem, I subcloned several RBL lines and tested their abundance of SGs. There was not much luck, either. I could not get a RBL line that has lots of SGs, and the abundance of SGs varies from batch to batch, too.

After all, I still could not establish the method to study SG fusion in giant excised patches. If someone is going to try this difficult project again (although not recommended), I will suggest him to semi-reconstitute the system using patch from RBL cells, and SGs from rat peritoneal mast cells. If it works, then try SGs from mouse peritoneal mast cells, and then liposomes filled with dopamine. None of the above is easy. What I can say is “good luck”...

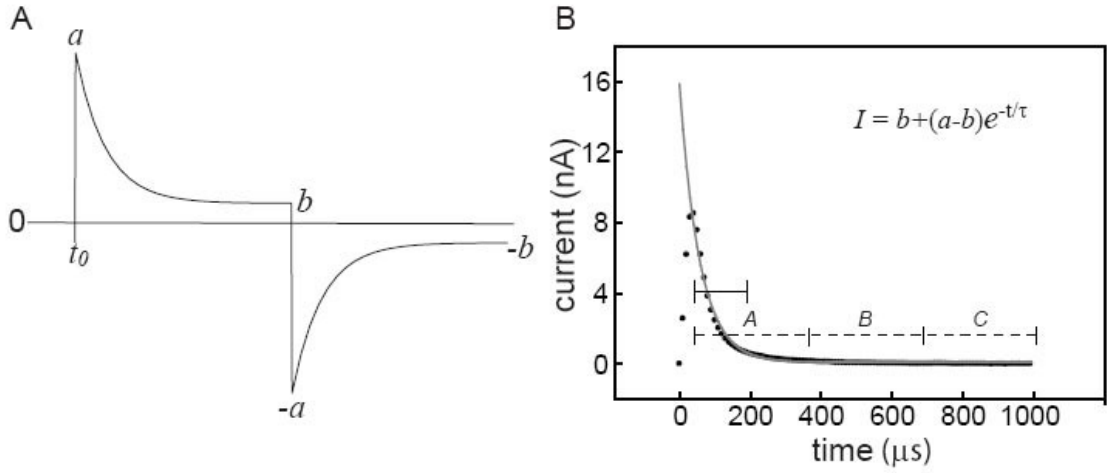


Figure 3.1. Method to determine whole-cell capacitance via square wave perturbation. Model current for whole-cell recording is shown in **A**. Peak current, *a*, and the projected steady-state current, *b*, were determined as described in the text. **B.** Half of the current from a real recording (dots) with the fitted exponential function used to determine cell parameters is shown. The asymptote of current was determined using the averages of three equally spaced data sections (dashed sections) according to Eq. 3.2, given in Chapter 3 and 4. The asymptote was subtracted and the data range from the peak to a point located at ~3τ, estimated as peak current times e^{-3}, was used to determine the exponential constants via linear regression of the log of the decaying current transient (solid line).

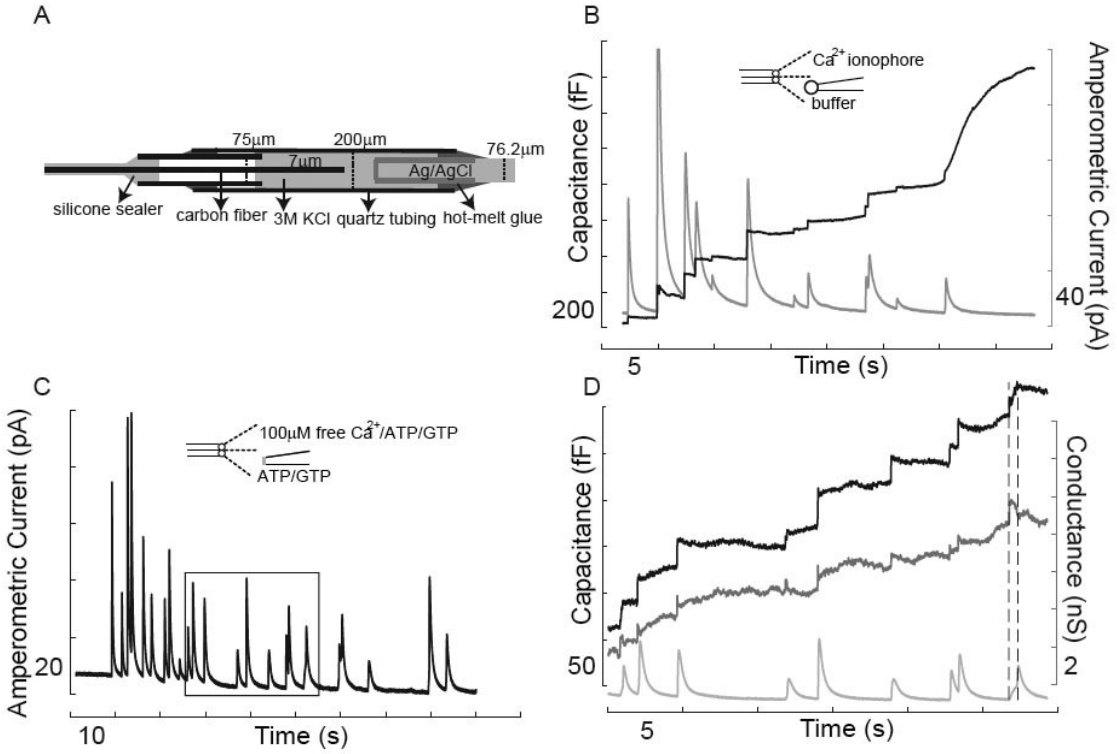


Figure 3.2. Amperometric and capacitance measurement in RBL cells. A. Schematic illustration of the intra- patch pipette carbon electrode. Carbon electrodes were prepared to allow facile insertion and manipulation in the patch pipette holder employed. **B.** In the cell-attached configuration, 2 µM of calcium ionophore, A23187, triggers profuse exocytosis. Two distinct vesicle pools are observed. One is the secretory granule (SG) pool with large-amplitude capacitance steps and amperometric spikes upon stimulation. The other pool (at the end of the trace) contains vesicles of much smaller size that do not release serotonin (non-SG pool). **C.** Amperometric recording of SG fusion in a ~20 µm (diameter) excised patch. **D.** Expandsion of **C**. In most cases, SGs are lost from membrane patches during the excision procedure. Occasionally, when SGs are preserved, fusion gives rise to capacitance steps of tens of fF indicating that the diameter of the granules is close to micrometer range. A typical fusion event with fusion pore dilation is marked between two broken lines. A gradual increase of capacitance, transient increase of conductance, and the amperometric foot-signal is observed. Here and in all subsequent figures numbers given between axis ticks indicate the tick interval.

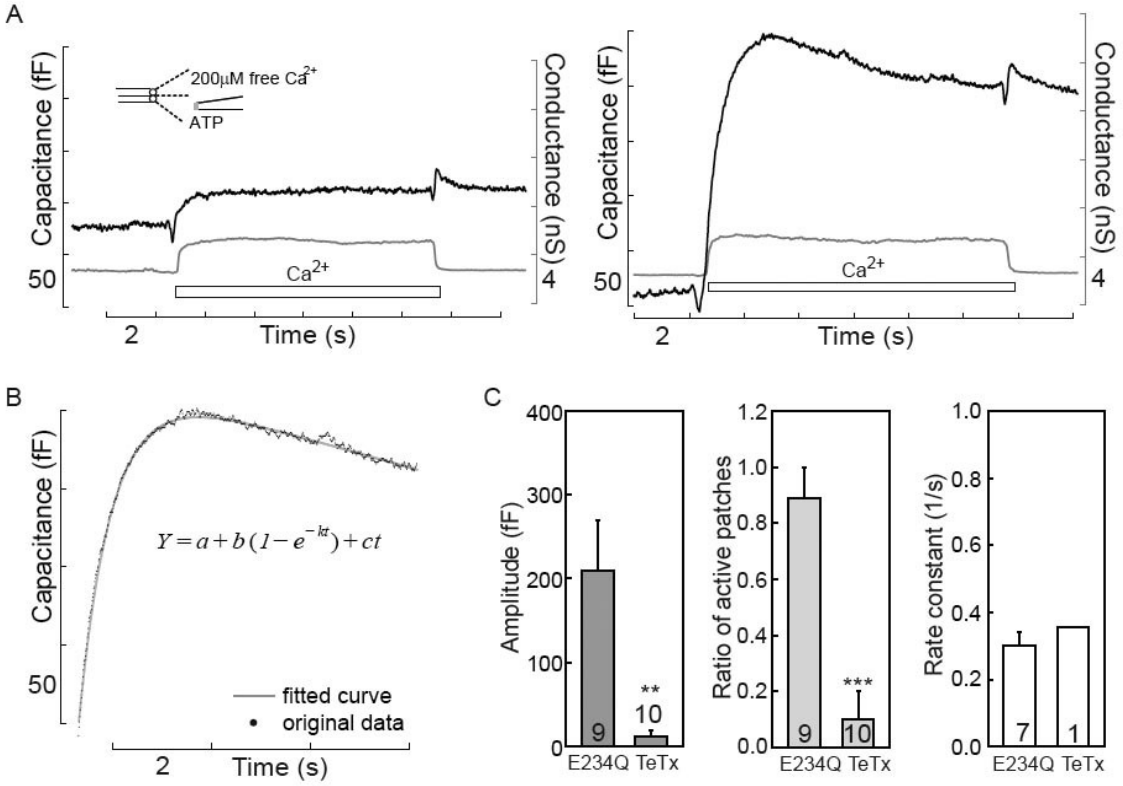


Figure 3.3. Non-SG fusion is SNARE-dependent. A. Two typical recordings from excised patches are shown. In the left panel, the amplitude is smaller than 50 fF, which is close to the artifact caused by moving the patch. Therefore, this patch is designated 'inactive'. In the right panel, robust non-SG fusion (> 50 fF) is observed in an 'active' patch. **B.** The same trace is fitted to a mono-exponential function (rate constant, '*k*') with a linear 'creep' component ('*c*'). **C.** Incubation of patches with 200 nM tetanus toxin light chain (TeTx) for 2 min blocks fusion while the mutant toxin (E234Q) has no effect, indicating that non-SG fusion is SNARE-dependent.

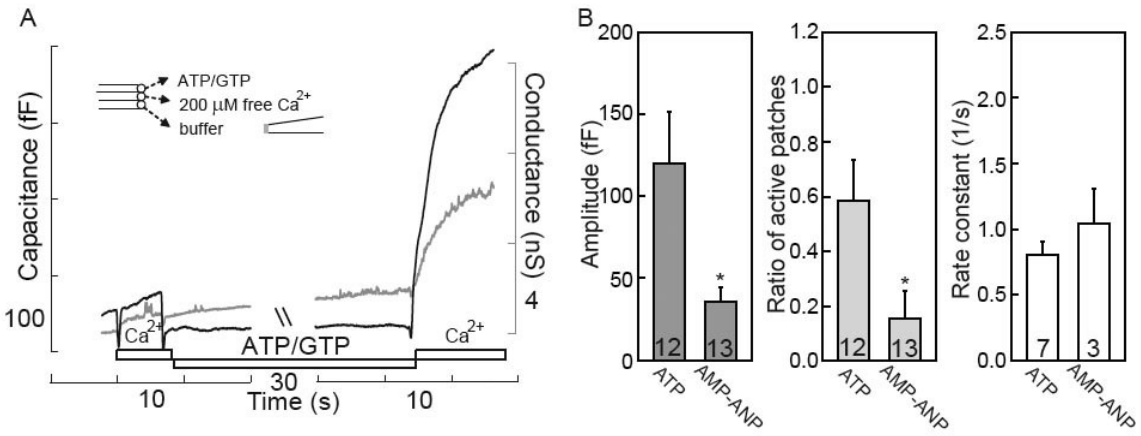


Figure 3.4. ATP hydrolysis is required for supporting non-SG fusion. A. Without ATP, Ca^{2+} application fails to trigger exocytosis in this patch, but exocytosis was restored by placing the patch in ATP/GTP-containing solution for an additional minute. **B.** AMP-PNP can not preserve fusion, indicating that the hydrolysis of ATP is required to support non-SG fusion.

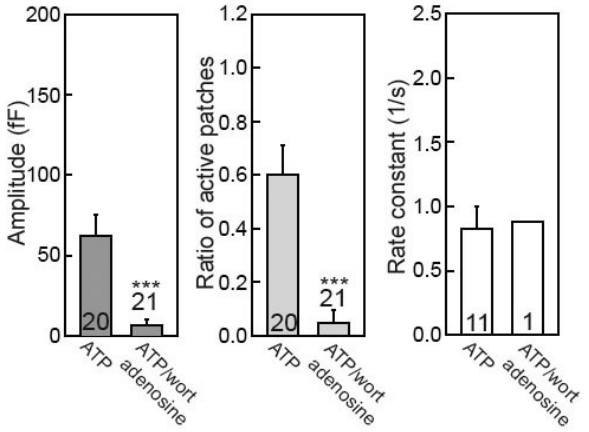


Figure 3.5. Non-SG fusion in excised patches is wortmannin/adenosine-sensitive. Incubation of patches with the PI-kinase inhibitors, wortmannin (wort; 4 μM) and adenosine (0.5 mM), significantly decreases the average fusion magnitudes and the ratio of active patches.

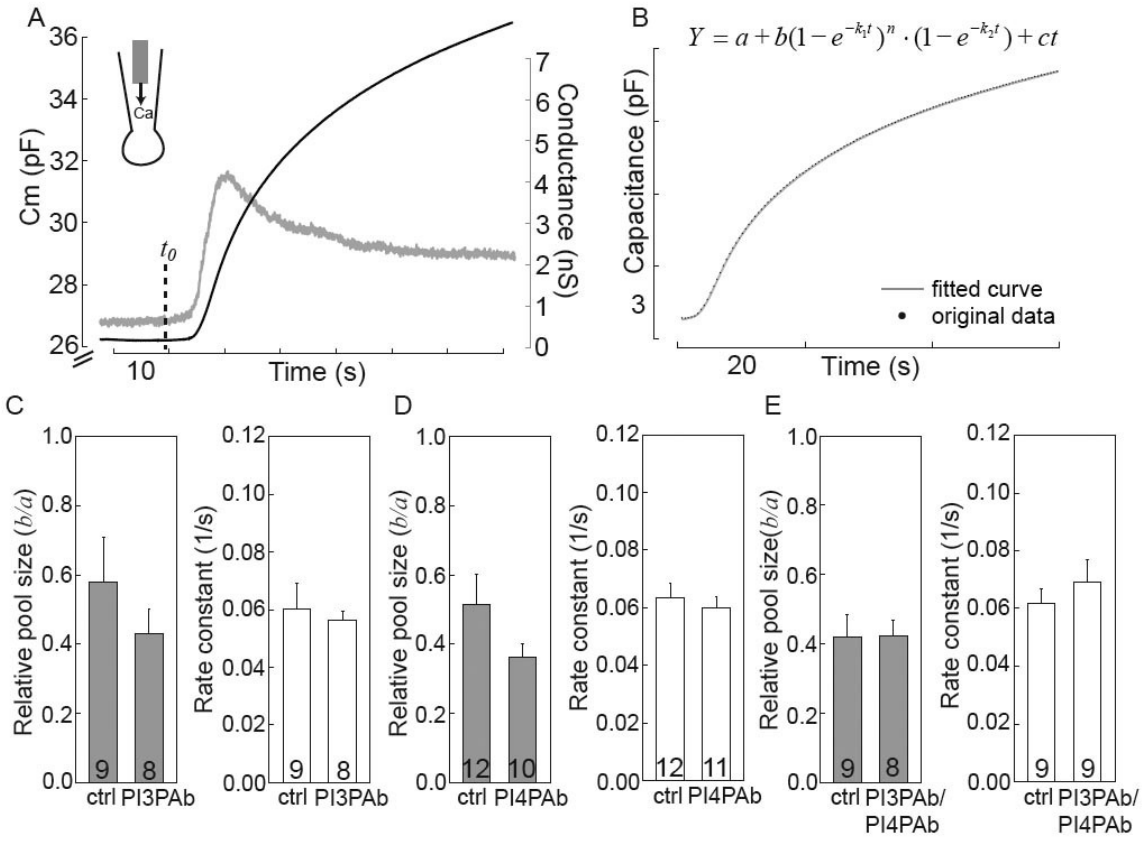


Figure 3.6. Non-SG fusion is not blocked by antibodies against PI(3)P and PI(4)P. In whole-cell recording, antibodies (1:100 dilution) were added to the cytoplasmic (pipette) solution, which was dialyzed into cells for 5 min before Ca^{2+} was infused to trigger fusion. **A.** A typical record from an RBL cell. The Ca^{2+}-activated conductance rise was used to define the t_0 point. **B.** The rise of capacitance was well described by the delayed-mono-exponential function given in the figure. The variable, $k2$, is the rate constant used for statistical analysis. The number of the data points was reduced to highlight the fitted curve. Antibodies against PI(3)P (**C**), PI(4)P (**D**), and both (**E**) fail to block non- SG fusion.

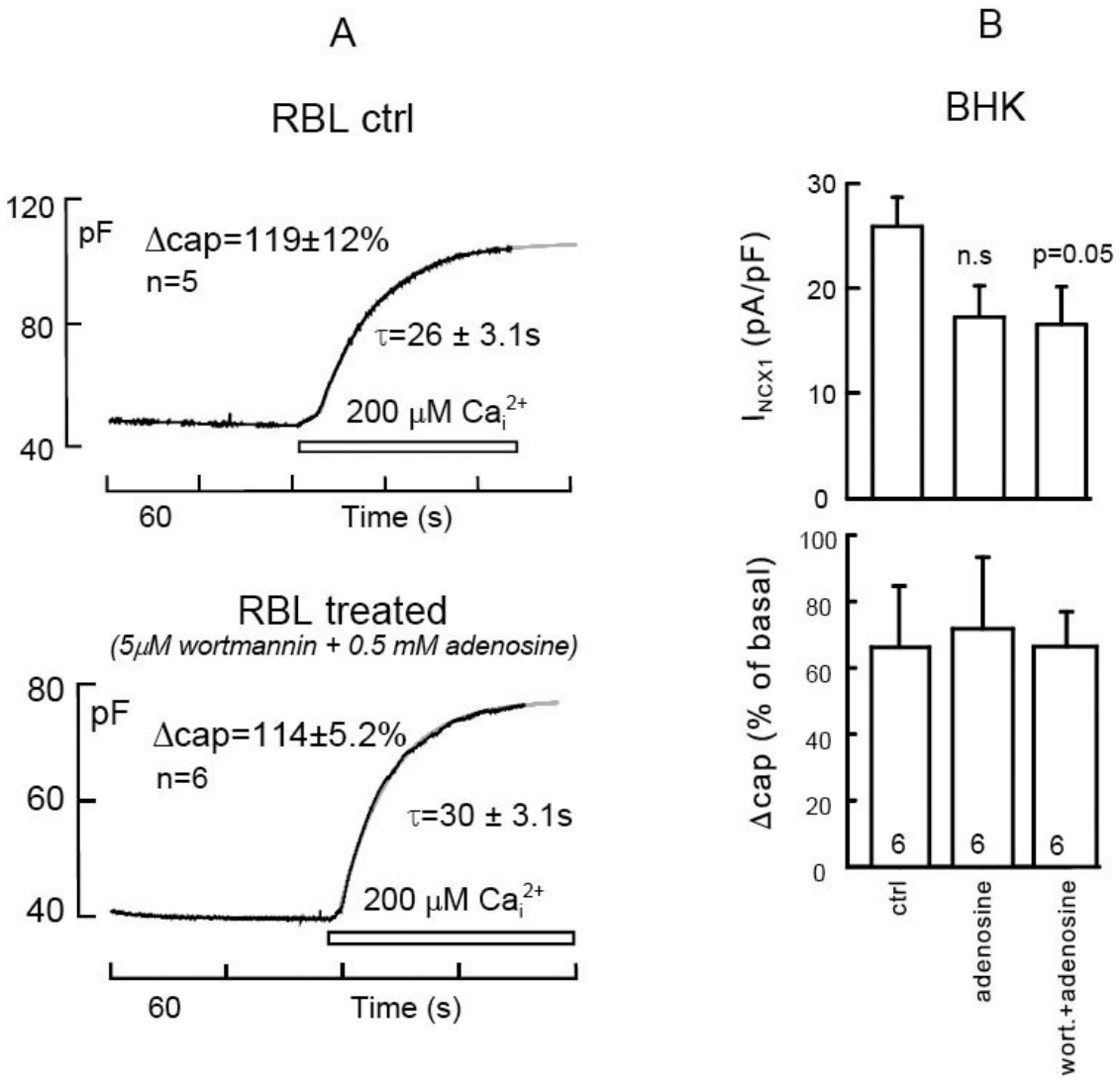


Figure 3.7. Non-SG fusion in whole-cell recording is wortmannin/adenosine-insensitive. A. Typical whole-cell capacitance records and composite statistics for RBL cells during cytoplasmic infusion of 200 μM Ca with control cytoplasmic solution (upper) and with 5 μM wortmannin and 0.5 mM adenosine (lower). Differences are not significant. **B.** Exchange current densities (upper) and capacitance responses (lower) for BHK cells in which membrane fusion was activated by outward Na/Ca exchange current with control cytoplasmic solution (left bar graph), with 0.5 mM adenosine (middle bar graph), and with 5 μM wortmannin and 0.5 mM adenosine (right bar graph) are shown.

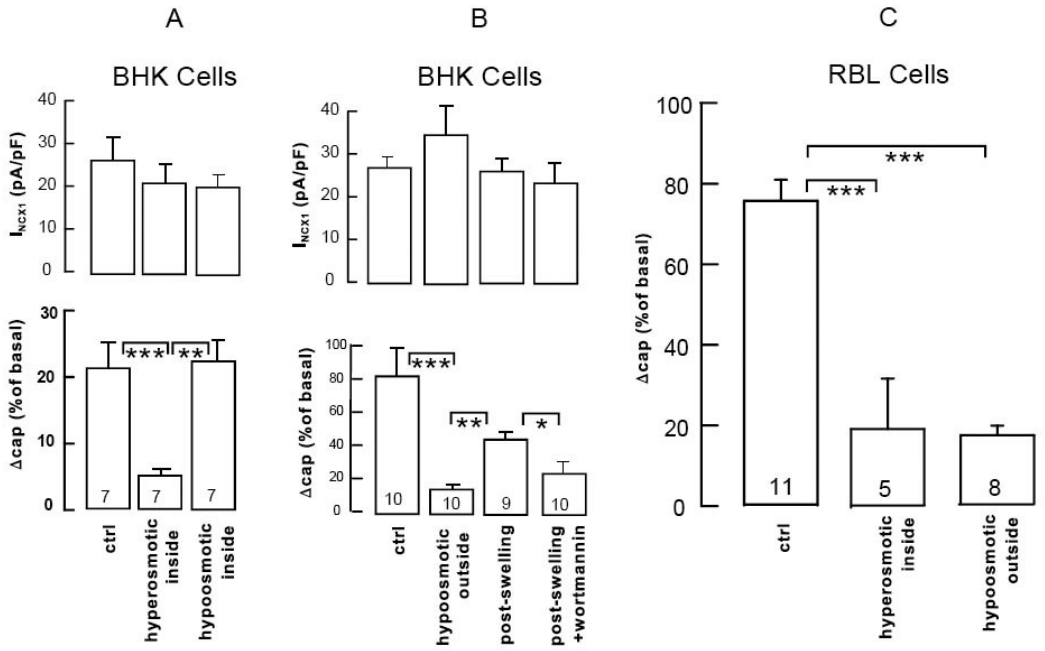


Figure 3.8. Non-SG fusion in whole-cell recordings is strongly inhibited by cell swelling. A & B. Bar graphs give exchange current densities (upper) and capacitance responses (lower) for two batches of BHK cells in which membrane fusion was activated by outward Na/Ca exchange current **A.** Effects of cytoplasmic osmolarity on fusion responses. The left bar graphs give results for isoosmotic solution, the middle bars for cytoplasmic solution with 200 mM sucrose, and the right bars for cytoplasmic solution diluted 30% with distilled water. **B.** Effects of extracellular osmolarity on fusion responses. From left to right the bar graphs are with (1) standard extracellular solution for 2 min after cell opening, (2) standard extracellular solution with the NMG-aspartate concentration reduced by 80 mM for 2 min after cell opening, (3) standard solution reapplied for 2 min after applying hypoosmotic solution for 2 min, and (4), as in (3) with 5 μM wortmannin in all cytoplasmic solutions. **C.** Capacitance responses of RBL cells for pipette perfusion of cytoplasmic solution with 200 μM free Ca. The left bar graph indicates the response magnitude for control cells, the middle graph for cells swollen with hyperosmotic cytoplasmic solution (200 mM sucrose) for 2 min, and the right bar for cells swollen with extracellular solution in which the NMG concentration was reduced by 80 mM.

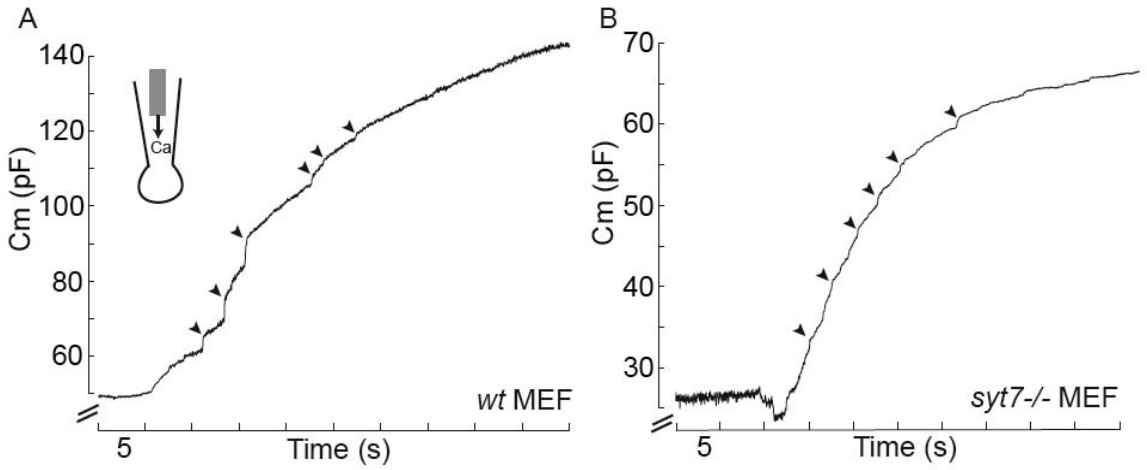


Figure 3.9. Synaptotagmin VII is not required for non-SG fusion. Whole-cell recordings from wildtype (*wt;* **A**) and *syt7* knockout (**B**) mouse embryonic fibroblasts (MEFs). Fusion remains robust in *syt7* knockout MEFs. Unlike results with RBL cells, non-SG fusion in MEFs often involves fusion of large vesicles (arrowheads).

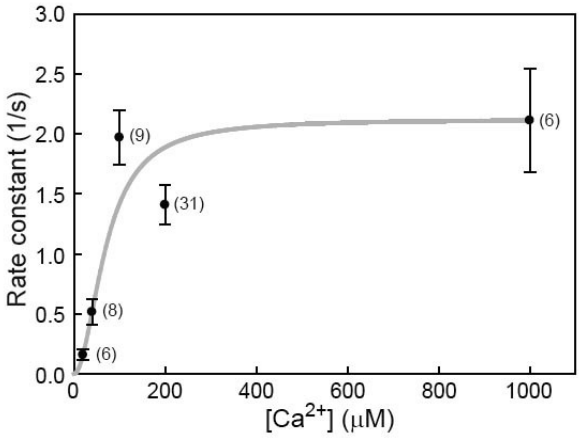


Figure 3.10. Ca-dependence of non-SG fusion in excised patches from RBL cells. The concentration-response data are best described by a Hill equation with a slope coefficient of 2.0, a K_m of 71 μM, and a maximal rate constant of 2.1 s^{-1}.

Table 3.1. Effects of various reagents on non-SG fusion in excised patches

Paired control Testing condition	Amplitude (fF) mean±SEM (n)	Ratio of active patches [a] mean±SEM (n)	Rate constant [a,b] (1/s) mean±SEM (n)
TeTx E234Q	209.2±60 (9)	0.89±0.11 (9)	0.3±0.04 (7)
TeTx WT	10.9±8.4 (10) **	0.1±0.1 (10) ***	0.36 (1)
ATP/GTP	160.8±42.2 (18)	0.67±0.11 (18)	1.52±0.27 (12)
NEM/ATP/GTP	138.5±43.7 (18)	0.5±0.12 (18)	1.01±0.21 (9)
ATP	119.8±31.8 (12)	0.58±0.15 (12)	0.8±0.11 (7)
AMP-PNP	35.3±9.4 (13) *	0.15±0.1 (13) *	1.04±0.27 (3)
Mg^{2+} buffer	53.3±19.9 (10)	0.3±0.15 (10)	1.12±2e-3 (2)
EDTA	168.6±70.7 (10)	0.8±0.13 (10) *	1.5±0.34 (8)
EDTA 1min	221.3±106.2 (5)	0.8±0.2 (5)	0.72±0.11 (4)
neomycin/EDTA	102.4±38.5 (5)	0.8±0.2 (5)	0.47±0.1 (3)
FVPP 1min	267.3±61 (8)	1 (8)	0.75±0.29 (8)
PIP_2Ab/FVPP	118.3±25.6 (7)	0.86±0.14 (7)	1.12±0.3 (6)
FVPP 2min	78.2±39 (6)	0.5±0.22 (6)	0.32±0.09 (2)
PIP_2Ab/FVPP	128.9±40.4 (6)	0.67±0.21 (6)	0.68±0.17 (4)
Mg^{2+} buffer	93.1±44.2 (13)	0.38±0.14 (13)	1.08±0.35 (5)
neomycin	71±22.9 (14)	0.29±0.13 (14)	0.62±0.21 (5)
Mg^{2+} buffer	103.1±47.3 (14)	0.43±0.14 (14)	0.92±0.19 (6)
PIP_2Ab	92.6±46.8 (14)	0.43±0.14 (14)	1.17±0.13 (5)
ATP	76.6±16.9 (14)	0.64±0.13 (14)	1.06±0.23 (9)
staurosporine/ATP	50.3±13.8 (15)	0.47±0.13 (15)	1.25±0.34 (7)
ATP	62.2±13.5 (20)	0.6±0.11 (20)	0.83±0.17 (11)
wort/adeno/ATP	6.9±3.1 (21) ***	0.05±0.05 (21) ***	0.88 (1)
ATP	47.3±26.4 (8)	0.38±0.18 (8)	1.04±0.32 (3)
LY294002/ATP	159.1±70.9 (7)	0.57±0.2 (7)	1.49±0.19 (4)
ATP	104.9±75.1 (4)	0.5±0.29 (4)	1.87±0.51 (2)
PI-TP/ATP	202±50.4 (4)	1 (4)	1.28±0.46 (4)

TeTx: tetanus toxin light chain, 200 nM; NEM: *N*-ethylmaleimide, 1mM; AMP-PNP: 2 mM; neomycin: 500 μM; PIP_2Ab: 1:50; staurosporine: 200 nM; wort: wortmannin, 4 μM; adeno: adenosine, 0.5 mM; LY294002: 100 μM; PI-TP: PI-transfer protein, 140 μg/ml; *a*: counts patches with amplitude >= 50 fF; *b*: outliers are removed using Grubbs' test; * P<0.05; ** P<0.01; *** P<0.001.

Chapter 4. Software and algorithm development

4.1 Introduction

There are several ways to measure the membrane capacitance of a cell. Depending on the method used to extract the information from the current waveform, they generally fall into one of the two categories: time-domain method, frequency-domain method (Lindau and Neher, 1988). The time-domain method uses square pulses to stimulate the cell and extract the information by fitting the resulting current transient with a decaying exponential function (Thompson et al., 2001). The obtained parameters are then used to calculate membrane capacitance (*Cm*), membrane resistance (*Rm*), and access resistance (*Ra*). In the frequency-domain method, a technique called phase-sensitive detection (PSD) is used. By using PSD, superior signal-to noise ratio is achieved.

Two different approaches have their own advantages and drawbacks over the other. The time-domain method (I used to call it "square-wave algorithm", SQA) is much noisier than the PSD especially when the time constant (τ) is fast. Nevertheless, the curve-fitting procedure is not always working and takes a lot of CPU power. A beautiful exponential decay is usually required for the calculation. However, unlike PSD, SQA gives you the absolute values of *Cm*, *Rm* and *Ra*, and most importantly, the calculation is independent of the phase angle which might be changing during the experiment (will be discussed later). The PSD uses a reference wave with a shift of "phase angle" to extract the information. Thus, setting the angle at different values result in different readouts for the same input. In addition, by using two references, signals proportional to *C* and *G* are

exported. Since no information about *Ra* is obtained, the absolute value of *C* and *G* cannot be calculated. The advantages of PSD are its superior noise performance and the ease of computer implementation.

Depending on the recording configuration, either of the methods might be better than the other in that specific case. The phase angle of the PSD depends on the clamping time constant, τ, and the oscillating frequency of the wave (Santos-Sacchi, 2004). The τ equals to *RC* where *R* is *RaRm/*(*Ra*+*Rm*). That is, if *Ra* and/or *Rm* change, the phase angle will change. For experiments using excised patches, since *Ra* is almost non-existing, charging of the membrane is superfast (i.e. time constant is very small), and PSD apparently is the best method to use. Actually, PSD is the only way to measure *Cm* in excised patches because the current transient is too fast to be fitted by SQA. Potentially the triangular-wave could be used to calculate absolute *Cm* in excised patches, however, since the actual *Cm* is always contaminated by the stray capacitance from the electrode, knowing the absolute *C* is actually not quite meaningful.

If PSD is to be used for whole-cell recording, it is important to keep in mind that the phase angle might be changing during the recording. It might be due to the reseal of the opening, changing of the seal resistance, or opening of the ion channels on the membrane (e.g. Ca^{2+}-activated chloride channels) etc. SQA might be a good choice if the τ is not too fast. For cells like cardiomyocyte, the clamping τ can be as long as several hundreds of μs. In this case, high resolution PSD could not be achieved because slow oscillating frequency is used (in order to charge the membrane properly). In contrast, slower τ

usually gives better curve-fitting result using SQA and the noise will be smaller as a consequence.

PSD using dual frequencies provides a solution for overcoming the drawbacks of single-frequency PSD (Barnett and Misler, 1997; Santos-Sacchi, 2004). The algorithm is generally based on the fact that capacitance signal is proportional to the oscillation frequency but not the conductance signal. There are computer programs available which adopt the dual-frequency principle, like PULSE (www.instrutech.com), and jClamp (www.scisoftco.com) etc. The single-frequency software lock-in amplifier can also be found online: pClamp (www.moleculardevices.com) (I don't know if it's using single- or dual-frequency... We don't have pClamp in the lab), NI lock-in amplifier (www.ni.com), and UTiLIA (mrflip.com/papers/LIA) etc. So, software packages for measuring capacitance are commercially available and some of them are even free... Why did I still develop this program? To make a long story short, it is because: 1. I do not have any commercially available software in the lab; 2. my mentor prefers to use MATLAB; 3. I love to make my ideas come true; and 4. It turns out that the program is more than just a lock-in amplifier. If you are not interested in this history, please skip the next paragraph without feeling guilty.....

Here is the actual story... When I jointed the lab, we were still using the chart recorders... Although LabView from National Instrument was already a data acquisition and analysis platform, my mentor was never happy about the graphic functions of it. Why didn't we try pClamp from Molecular Devices or PULSE from HEKA? It's clear that to use PULSE, the entire lab had to switch from Axopatch to HEKA amplifiers; for pClamp,

I don't know the actual reasons... but I guess my mentor wanted to have more control over the functions of the software, so he decided to develop a program based on the MATLAB platform. The first acquisition software in the lab is called program2 developed by Chengcheng Shen, who became a good friend of mine after he jointed the lab. The program2 served as a plain data recording software and everyone was happy about it. However, human beings always want more... At that time, I was trying to do carbon fiber amperometry, and one of the desired function is the digital filtering. To make real-time digital filtering, the core of program2 had to be redesigned. In addition, I also wanted more functions like pulse generation and some changes of the graphic-user interface (GUI) of program2. Everyone has his own business, I could not always ask Chengcheng to do something for me. As a fearless biologist, I decided to learn MATLAB and design new program which satisfies my needs. The first developed program is called Capmeter 1 which still serves as a plain data recorder but has the ability to do real-time digital filtering (will be described in later sections). This program is never being popular in the lab probably because people tend to fix on what they used to use... As the time passed-by, two out of three lock-in amplifiers in the lab showed signs of malfunctioning. To save money for my mentor, I decided to add PSD to my program, which is quite easy as mentioned earlier. The resulting Capmeter versions started to be accepted in the lab, and the current version is Capmeter6v3 as of Feb. 2008. In addition to PSD, Capmeter6v3 uses two SQA methods to extract the information. The I-SQA fits the current transient directly while the Q-SQA integrates the current first and then fits the resulting time-charges trace.

To synchronize analog input and output in the MATLAB, one could either do it at the very beginning of the acquisition, or do it repetitively. If the first method is to be used, the MATLAB function *peekdata* rather than *getdata* has to be used because the *Logging* property of the analog input remains ON during the acquisition and data can not be extracted. As of its name, the *peekdata* function does not extract the data from the MATLAB Engine and the returned data may be missed or repeated (please refer to the function references of MATLAB). The raw data have to be retrieved and processed again after the acquisition, and there is always a risk to overload the RAM during recording, too. Capmeter uses the second approach. The timing of analog output and input is synchronized by embedding a 1 mV trigger signal (if the command sensitivity is 20 mV/ V) on top of the generated sine or square wave every 10 ms (if digitized at 100 Hz). The data are extracted from the MATLAB Engine between triggers and then processed (digitized) by Capmeter. Since data are extracted piece by piece, problems mentioned above are avoided. In the following sections, I will show you how to use the programs and explain how the programs work.

4.2 Algorithms for capacitance measurement

Phase-sensitive detection

The lock-in amplifier is invented by physicist Robert H. Dicke at Princeton University (source: wikipedia.org). PSD gives superior signal-to-noise performance because it only detects signals oscillating at a specific frequency. Since from the Fourier-theorem's point of view, the background noise can be considered as a combination of periodic waves oscillating at the entire frequency spectrum, noise oscillating at frequencies other than the designated one is eliminated, and thus, superior noise performance is achieved.

Mathematically, considering a signal is oscillating at an angular speed of ω_s, a phase angle of θ_s, and with amplitude of V_{sig}, the signal can be expressed as

$$V_{sig}\sin(\omega_s t+\theta_s) \qquad (4.1).$$

The product of the signal and a reference wave at an angular speed of ω_r, a phase angle of θ_r,

$$\sin(\omega_r t+\theta_r) \qquad (4.2), \text{ is}$$

$$\frac{1}{2}V_{sig}[\cos((\omega_s-\omega_r)t+\theta_s-\theta_r)-\cos((\omega_s+\omega_r)t+\theta_s+\theta_r)] \qquad (4.3).$$

If ω_r equals to ω_s (i.e. signal oscillates with the command voltage at the same frequency), Eq. 4.3 becomes a mixture of DC and AC components. After removing the AC component using low-pass filtering, the double of the DC component of Eq. 4.3 is

$$X=V_{sig}\cos(\theta_s-\theta_r) \qquad (4.4).$$

After proper adjustment of θ_r so that it equals to θ_s (i.e. the conductance component oscillates in-phase with the command voltage), X in Eq. 4.4 represents the actual magnitude of the signal (the conductance component). By adding $\pi/2$ to θ_r (which is the oscillating phase angle of the capacitive component), the double of the DC component of Eq. 4.3 is

$$Y = V_{sig} \sin(\theta_s - \theta_r) \qquad (4.5).$$

Again, if θ_r is adjusted properly, Y in Eq. 4.5 represents the actual magnitude of the capacitive component. For signals oscillating at angular speed other than ω_r (i.e. background noise), the DC component of Eq. 4.3 is basically zero (beacuse $\omega_s - \omega_r \neq 0$), thus they are sequestered after low-pass filtering.

It is also very helpful to view how PSDs work geometrically. First, let's consider only the conductance component (G) for now. In Fig. 4.1A, assuming that the phase angle is not properly adjusted (i.e. $\theta_s - \theta_r \neq 0$), according to Eq. 4.4, the readout of PSD1 is the projection of G on the x-axis (X), and G also contaminates the readout of PSD2 (Y), which is $\pi/2$ away from PSD1 (Eq. 4.5). To adjust the reference phase angle θ_r, it is just like rotating the coordinates by an angle of $\theta_s - \theta_r$, as shown in Fig. 4.1B. After the adjustment, X reflects the actual amplitude of G and nothing contaminates PSD2.

When θ_r is properly adjusted, PSD1 and PSD2 give the actual G and C, respectively. The mixture of the two components is a new vector, with an amplitude (R) of $(G^2+C^2)^{1/2}$ and a phase angle (θ) of $tan^{-1}(C/G)$ (Fig. 4.1C). It is important to note, however, by knowing R and θ does not mean that you can extract the actual G and C. There are

infinite ways to get a vector with R and θ (Fig. 4.1D, three of them are shown), and the combination of G and C is only one of them. The hardware SR830 DSP dual phase lock-in amplifier (Stanford Research Systems, Sunnyvale, CA) calculates and outputs the values of R and θ, too. The user's manual of SR830 is also a very good introduction of PSD, which is at http://www.thinksrs.com/downloads/PDFs/Manuals/SR830m.pdf.

Phase-sensitive detection – online calculation of the phase angle

To test whether the current θ_r is valid or not, one can change the amount of compensated capacitance from the patch clamp, and see if the change (ΔC) is solely reflected on the readout of PSD2. If θ_r is not correct, changes of capacitance compensation result in signal changes both on PSD1 (X-X_0) and PSD2 (Y-Y_0), as shown in Fig. 4.2. The correct phase angle, θ_r', can be calculated by rotating the PSD1-PSD2 coordinate by an angle of α, given that α is $tan^{-1}((X\text{-}X_0)/(Y\text{-}Y_0))$ and X_0 (Y_0) and X (Y) represent values before and after changing capacitance compensation, respectively. The equation can be expressed as

$$\theta_r' = \theta_r - \arctan\left(\frac{X - X_0}{Y - Y_0}\right) \qquad (4.6).$$

It is important to keep in mind, however, that this method is valid only when the time constant of capacitance compensation is adjusted properly, else the changes will not be reflected at the actual phase angle. For excised patches, because Ra is almost non-existing, τ is superfast as mentioned in the Introduction. In this case, the knob of 'fast component' compensation should be used, and the shortest time constant setting is

selected. For whole-cell recording, the 'series resistance' knob on the 'capacitance compensation' panel of the patch clamp has to be adjusted properly. To do so, the only way I know is to play with the knobs and see at which setting the cell capacitance is compensated properly.

Phase-sensitive detection – offline adjustment of the phase angle

As introduced previously, when the reference phase angle θ_r is not set properly, X and Y may not reflect the actual G and C. If it is the case, an offline adjustment of the reference phase angle is desired. To do so, one can rotate the PSD1-PSD2 coordinate with a desired phase shift (α) to reconstruct a new set of X and Y. Considering that what the lock-in amplifier sees are two orthogonal vectors (X and Y) that compose a vector with R and θ (Figs. 4.3A). By rotating the PSD1-PSD2 coordinate (Fig. 4.3B,C), the new PSD1 readout (X_{adj}) is the sum of the projections of the original X and Y on the PSD1 axis, which is $X\cos(\alpha)$ and $Y\sin(\alpha)$, respectively. Again, the new PSD2 readout (Y_{adj}) is the sum of the projections of the original X and Y on the PSD2 axis, which is $-X\sin(\alpha)$ and $Y\cos(\alpha)$, respectively. The equations are summarized below,

$$X_{adj} = X\cos(\alpha) + Y\sin(\alpha) \qquad (4.7)$$

$$Y_{adj} = -X\sin(\alpha) + Y\cos(\alpha) \qquad (4.8).$$

Offline adjustment of phase angle is particularly useful for whole-cell records. As mentioned previously, phase angle might shift if the clamping time constant shifts. It is usually caused by changing of Ra, Rm, and/or Cm, which happen all the time during whole-cell recordings.

Phase-sensitive detection – computer implementation

The implementation of PSD is relatively simple as mentioned in the Introduction. In Capmeter, the program times the acquired current in a given time interval with either an in-phase or an orthogonal reference wave. The product is then averaged to get the DC signal, and the 2DC value is assigned to PSD1 or PSD2. An example is shown in Fig. 4.4, and the actual codes are shown below.

MATLAB codes for setting references

```
%--- Reference wave calculation
function Refcalc(FH)
handles = guidata(FH);
PPS = (handles.aiSR/(handles.PSDfreq*1000)); %points per sine wave
L = floor(handles.aiSamplesPerTrigger/PPS)*PPS; %to make sure that the
DC noise can be cancled
T = (L-1)/handles.aiSR;
P = handles.PSDphase*pi/180;
F = handles.PSDfreq*1000;
handles.PSDref = (sin(linspace((P+(pi/2)),((P+(pi/2))+(2*pi*F*T)),L)))';
handles.PSD90 = (sin(linspace((P+pi),(P+(pi*(1+(2*F*T)))),L)))';
guidata(FH,handles);
```

Every time when you change the oscillating frequency and/or adjust the phase angle, the program will call the function *Refcalc* to adjust the reference waves (*Ref1* and *Ref2* in Fig. 4.4). Since the program uses simply averaging to extract the DC component, it is important to make sure that the time span of the manipulated data points is the multiple of the period of a single sine wave. Variables *PPS* and *L* above ensure that it is the case. Notice that the last time point represented by the reference wave is (*L-1*)/*handles.aiSR*, but not *L*/*handles.aiSR*, where *handles.aiSR* is the acquisition speed of the analog input (points/s). Time point *L*/*handles.aiSR* is the first time point of the next processing cycle.

The phase shift (*P* or *handles.PSDphase* above) is added into the calculation of the reference waves (*handles.PSDref* and *handles.PSD90*). You might have noticed that the

first point of *handles.PSDref* is sin($P+\pi/2$), but not sin(P) as introduced in Eq. 4.2 in earlier section. It is because the acquisition of the signals is initiated by a trigger signal added on top of the peak of the command sine waves, the first acquired point is resulted from command potential $Vsin(\pi/2)$ but not $Vsin(0)$, which is zero.

The multiplication of the current with the reference waves is carried out in one of the subfunctions in *CapEngine4.mexw32*, which is written in C.

```
C codes for PSD multiplication and averaging
void PSD(double *data,double *ref,int Mref,int L,int ppch,double
*output)
{
    int i,j;
    double *A;
    L +=1; //+1 is the NaN
    A = (double *)mxMalloc(Mref*sizeof(double));
    for(i=0;i<ppch;i++) //data point index
    {
        for(j=0;j<Mref;j++) //data*ref
        {
            A[j] = data[(i*L)+j]*ref[j];
        }
        output[i] = 2*Cmean(A,Mref);
    }
    mxFree(A);
}
```

It is clear from the codes that the function allocate the memory for array A first, and then assign A the product of the current (*data*) and reference wave (*ref*) in a element-by-element manner. The DC component is calculated by averaging the entire array A, and the 2DC value is the output of the function.

Square-wave perturbation – based on fitting the current transient

Equations in this section are derived by my mentor, Dr. Hilgemann and I, so I use "we" rather than "I" in this section. The whole-cell patch clamp can be considered as an

R-C circuit. Current flowing through the electrode is determined by access resistance, *Ra*, membrane capacitance, *Cm*, and membrane resistance, *Rm*. Equations for extracting cell parameters from peak current, steady-state current and time constant, have been introduced in Chapter 3. In this section, I will discuss and derive the equations again in more detail, starting from the estimation of the steady-state current.

Current flowing through an R-C circuit can be described as an exponential function,

$$I = b + (a-b)e^{-t/\tau} \qquad (4.9),$$

where a, b, and τ are the peak current, steady-state current, and time constant, respectively (Fig. 4.5A, B). To estimate b, one first takes three equally-spaced points, A, B, and C, from the decaying curve. If the distance between A and B is Δ, then

$$A = b + (a-b)e^{-t_1/\tau} \qquad (4.10)$$

$$B = b + (a-b)e^{-(t_1+\Delta)/\tau} = b + (a-b)e^{-t_1/\tau}\cdot e^{-\Delta/\tau} \qquad (4.11).$$

By defining

$$m \equiv (a-b)e^{-t_1/\tau}, n \equiv e^{-\Delta/\tau} \qquad (4.12),$$

Equations can be re-written as

$$A = b + m \qquad (4.13)$$

$$B = b + mn \qquad (4.14)$$

$$C = b + mn^2 \qquad (4.15).$$

Solving Eq. 4.13, 14, 15 simultaneously gives Eq. 3.2, which is

$$b=\frac{B^2-AC}{2\mathrm{B}-A-C} \qquad (3.2).$$

However, as mentioned in Chapter 3, and as shown in Fig. 4.5B, we use the averages of section *A*, *B*, and *C* rather than using three single points. By doing this, the noise of the estimation is suppressed because of the averaging. To proof that Eq. 3.2 is still valid when the section averages are used, considering that the sum of the sections are,

$$\sum_{i=t_1}^{i=t_1+d} A_i = \sum_{i=t_1}^{i=t_1+d} (b+m_i) \qquad (4.16)$$

$$\sum_{i=t_1+\Delta}^{i=t_1+\Delta+d} B_i = \sum_{i=t_1}^{i=t_1+d} (b+m_i n) \qquad (4.17)$$

$$\sum_{i=t_1+2\Delta}^{i=t_1+2\Delta+d} C_i = \sum_{i=t_1}^{i=t_1+d} (b+m_i n^2) \qquad (4.18).$$

By defining

$$M \equiv \frac{1}{d} \sum_{i=t_1}^{i=t_1+d} m_i \qquad (4.19),$$

the section averages *A*, *B*, and *C* can be re-written just like Eqs. 4.13, 14, and 15, except that the *m* is replaced by *M*. Accordingly, the solution is still Eq. 3.2.

After subtracting the steady-state current, *b*, from the half pulse, the natural logarithm of the trace is

$$\ln(a-b)-t/\tau \qquad (4.20).$$

The slope and the intercept at time zero of Eq. 4.20 represent -1/τ and *ln*(*a*-*b*),

respectively. Since *b* is known, *a* is then obtained from the intercept. To estimate all three cell parameters, namely, *Ra*, *Rm*, and *Cm* from *a*, *b*, and τ, let's consider two situations, 1.when the membrane is not charged at all, and 2. when the membrane is fully charged.

In the first condition, since the membrane is not charged, current flowing through *Ra* is driven by a full voltage step of *2Vc*, where *Vc* is half of the command voltage. The relationship is written as

$$Ra = \frac{2\,Vc}{a+b} \qquad (4.21).$$

When the membrane is fully charged, the current flow through *Ra* first and then flow across *Rm* directly without charging the membrane. Current flowing through two resistors in series is described as

$$b = \frac{Vc}{Ra+Rm} \qquad (3.7).$$

The solution for *Rm* according to Eqs. 4.21, 3.7 is

$$Rm = \frac{Vc(a-b)}{b(a+b)} \qquad (4.22),$$

and membrane capacitance is always presented by Eq. 3.11, which is

$$Cm = \tau\left(\frac{1}{Ra}+\frac{1}{Rm}\right) = \frac{\tau(a+b)^2}{2\,Vc(a-b)} \qquad (4.23).$$

The above solutions for *Ra* and *Rm* are valid only when the membrane is properly charged. If the oscillation is too fast, that is, the step voltage changes its direction before the membrane is fully charged (which is one of the assumptions), the situation becomes

much more complex. We approach this issue by formulating the membrane potential during the fast oscillation. Considering that the command square pulse oscillates with a step size of $2Vc$, let's say from $-Vc$ to Vc for the ease of explanation for now, the relationship of membrane potential versus time is shown in Fig, 4.5C. Since the oscillation is too fast, the membrane potential can never reach its theoretical steady-state value (Vss), which is

$$Vss = \frac{VcRm}{Ra + Rm} \qquad (3.3).$$

Recalling that at steady state, the circuit is like two resistors in series, so part of the Vc is shared by Ra, and Vss is a fraction of Vc (Eq. 3.3). Assuming that the membrane potential is oscillating between $-fVss$ and $fVss$, where f is the fraction of Vss across the membrane at the end of the voltage step of duration, Δ, the relationship between membrane potential and time can be expressed as

$$V_{(t)} = -fVss + Vss(1+f)(1-e^{(-t/\tau)}) \qquad (3.4).$$

Solving for f with $t=\Delta$,

$$f = \frac{1-e^{-\Delta/\tau}}{1+e^{-\Delta/\tau}} \qquad (3.5),$$

and membrane voltage at the beginning of the voltage step is

$$V_{(0)} = \frac{-fVcRm}{Ra + Rm} \qquad (3.6).$$

From the steady state current,

$$b = \frac{Vc}{Ra + Rm} \qquad (3.7),$$

and the peak current,

$$a = \frac{Vc - V_{(0)}}{Ra} \qquad (3.8),$$

the solutions for *Ra*, *Rm* and *Cm* are

$$Ra = \frac{Vc(1+f)}{a+bf} \qquad (3.9),$$

$$Rm = \frac{Vc(a-b)}{b(a+fb)} \qquad (3.10),$$

$$Cm = \tau\left(\frac{1}{Ra} + \frac{1}{Rm}\right) = \frac{\tau(a+bf)^2}{Vc(a-b)(1+f)} \qquad (3.11).$$

When Δ is very long, *f* approaches one, and Eqs. 3.9, 3.10 and 3.11 approach exactly Eqs. 4.21, 4.22, 4.23, respectively.

The derivation of the above equations assumes that the command voltage is oscillating around zero volt. What if it is not the case (e.g. oscillating between *p* and *q* volt)? In this case, the command potential can be expressed as

$$Vc = O \pm Vsi \qquad (4.24),$$

where *O* is the average of *p* and *q*, and *Vsi* ('*si*' is pronounced as letter 'C') is half of the peak-to-peak amplitude. Since *O* is a constant potential, its resulting current is a DC current component. After subtracting the DC component from the total current, the resulting trace is identical to the one mentioned above. Thus, all equations derived above

are still valid. Note that the peak current in Eq. 3.8 is a, but in Eq. 4.21 is $(a+b)$. It is because the definitions for Vc are slightly different. In Eq. 3.8, Vc is the absolute value of the potential, and the baseline current is zero. In Eq. 4.21, Vc is a relative potential defined as half of difference between two peak potential, and in this case, the actual baseline current value is $-b$ for the raising potential in the fitting routine, so the peak current is $(a+b)$.

I-SQA – computer implementation

It seems easy to do the curve fitting and extract all cell parameters once we have the equations, however, it is not true. The real world is much more complex, and all kinds of unexpected errors and/or conditions will occur within the fitting routine if there is a chance for them. A tremendous amount of efforts have been made to find, test, and control these errors/conditions in order to make the fitting routine workable, and giving reasonable outputs.

The entire fitting routine is in a function called *SqCF* in *CapEngine4.mexw32*. The first step for the fitting routine is to find the positions of all the peaks. Since both the data acquisition and the command oscillation is at a fixed speed, it seems that the difference among peak indexes is a fixed constant (e.g. acquire at 100kHz and oscillate at 1kHz, one half pulse is represented by 50 points, theoretically), however, it is not true. Even when the acquisition speed is the multiple of the oscillating frequency, the distance between two half pulses is always one or two points away from each other (e.g. represented by 49 points or 51 points). So, the fitting routine has to locate the peak indexes from the trigger signals that are acquired from AI0.

C codes for peak detection

```
for(i=0;i<L;i++)
{
    dataA[i] = data[n*(L+1)+i];
    dataTrig[i] = trigger[n*(L+1)+i];
    dataTime[i] = time[n*(L+1)+i];
}
Cdiff(dataTrig,L,&diffabs);
for(i=0;i<(L-1);i++) {diffabs[i] = Cabs(diffabs[i]);}
Cfind(diffabs,1,threshold1,(L-1),&indexpeak,&fc);
if(fc!=0)
{
    for(i=0;i<fc;i++) {indexpeak[i] = indexpeak[i]+1;} //adjust the
index
}

Sp = fc-1; //remove the last curve, bcz it's usually incomplete
if(Sp < 1) {quality = 0;}//poor quality

if(quality)
{
    ...other fitting procedures...
}
```

Variable *dataTrig* is an array containing the command output recorded by AI0. The absolute difference between each points is calculated by *Cdiff* and *Cabs* (user defined, not in standard library), and the peak indexes are determined by *Cfind* using a threshold of 0.15 (*threshold1*). Variable *fc* represents number of peaks found, and because the last curve is usually incomplete according to my experience, it is removed from the data set. If no peak is found (*Sp*<1), variable *quality* is set to zero and other fitting procedures will be skipped.

The next step is to calculate the baseline (or middle line more precisely) of the current trace. This step is important because the DC current is not always (or rarely) zero. Considering that the seal becomes bad during the recording and the liquid junction potential is not compensated perfectly, or, ion channels open in an experimental condition, a DC current will flow into or out of the pipette for sure. In these cases the

middle line subtraction becomes critical.

```
C codes for middle line calculation
for(i=0;i<(fc-fmod(fc,2));i++)
{
    zerosum += dataA[(indexpeak[i])];
}
zeroline = zerosum/(double)i;
```

Variable *dataA* is an array containing raw current. The loops add equal numbers of positive and negative peaks together, and the average value is the middle line. According to the codes, it is important to keep in mind that data clipping shall be avoided because the middle line can not be determined correctly if it happens. In Capmeter 6, the PSD/I-SQA/Q-SQA pop-up menu (*handles.Cm*) changes its *BackgroundColor* to red if the calculated peak is larger than 10 volts using the following MATLAB codes,

```
MATLAB codes in function SqAlgo
try
    if (~isequal(get(handles.Cm,'ForegroundColor'),[0 0
0]))&&(~isnan(tau(1,1)))
        set(handles.Cm,'ForegroundColor','black');
    elseif (isequal(get(handles.Cm,'ForegroundColor'),[0 0
0]))&&(isnan(tau(1,1)))
        set(handles.Cm,'ForegroundColor','red');
    end
    if (isequal(get(handles.Cm,'BackgroundColor'),[1 0
0]))&&(max(peak(~isnan(peak))) < 10)
        set(handles.Cm,'BackgroundColor',get(handles.figure1,'Color'));
    elseif (~isequal(get(handles.Cm,'BackgroundColor'),[1 0
0]))&&(max(peak(~isnan(peak))) >= 10)
        set(handles.Cm,'BackgroundColor','red');
    end
end
```

The function *SqAlgo* is evoked periodically to calculate cell parameters using peak current, steady-state current, and time constant estimated by *CapEngine4*. In *SqAlgo*, if data clipping happens, the background color of the pop-up menu becomes red. In addition, if *CapEngine4* can not get the time constant (NaN, “not a number”, is assigned),

the *ForegroundColor* property of the list menu is set to red.

After subtracting the middle line value from the raw current, the fitting routine processes each curve individually. Starting from the steady-state current (asymptote) estimation, the codes are listed below.

C codes for asymptote estimation

```
signB = Csign(Cmean(&dataA[indexpeak[np]],SPC));
for(i=0;i<SPC;i++)
{
    dataB[i] = dataA[(indexpeak[np]+i)]*signB; //assign data and
correct the polarity
    dataBtime[i] = dataTime[(indexpeak[np]+i)]-dataTime[indexpeak[np]];
//set initial time to 0
}
Cfind(dataB,0,Cmax(dataB,SPC),SPC,&ftemp,&fc);
if(fc != 0) {lastmax = ftemp[(fc-1)];} //fc-1 is the last one.
else {lastmax = 10;}
if(lastmax > (SPC-12)) {lastmax = SPC-12;}
D = (int)floor((double)(SPC-lastmax)/3);
w = D;
sp1 = lastmax;
sp2 = sp1+D;
sp3 = sp2+D;
if((sp3+w) > SPC) {w = SPC-sp3;}

for(i=0;i<w;i++)
{
    s1 += dataB[(sp1+i)];
    s2 += dataB[(sp2+i)];
    s3 += dataB[(sp3+i)];
}
s1 /= w;
s2 /= w;
s3 /= w;

//get asymptote
if((2*s2-s3-s1) != 0) {asymp = (((s2*s2)-(s1*s3))/(2*s2-s3-s1));}
else {quality = 0;}

if(quality)
{
    procedures for peak and time constant estimation...
}
```

To correct the polarity of the curve, the routine multiplies the curve with the sign of its

average (variable *signB*). For asymptote estimation, the averages of three equally spaced sections are used as introduced in Eq. 3.2. Rather than starting from the first point to the last one, a variable called *lastmax* is introduced. As shown in Fig. 4.5B (dots), in the presence of a filter function, the peak is usually delayed and rounded. If the entire data range were used for asymptote estimation, apparently, the value of section *A* would be under estimated. Variable *lastmax* represents the last maximal point of the curve. I use the last maximal point because if data clipping happens (or when the seal is gone), many points around the real peak will be with the same value, which is 10 volts. Variable *SPC* stands for "samples-per-curve". If the range for asymptote estimation (*lastmax* to *SPC*-1) is less than 12 points (determined empirically), a point at *SPC*-12 is assigned to *lastmax*. The size of each section is then adjusted so that no memory leakage will occur. The asymptote is calculated using Eq. 3.2, and if the estimation cannot be done, variable *quality* is set to zero to skip the coming fitting procedures.

To calculate the peak current and the time constant, the routine uses Eq. 4.20 and linear regression.

```
C codes for peak and tau estimation
fladd1 = (firstmin-lastmax+1);
for(i=0;i<SPC;i++) {dataB[i] -= asymp;} //It is ln(a-b)
temp = (double *)mxRealloc(temp,fladd1*sizeof(double));
for(i=0;i<fladd1;i++) {temp[i] = dataB[(lastmax+i)];}
Cfind(temp,-10,0,fladd1,&ftemp,&fc);
if(fc != 0) {firstmin = lastmax+ftemp[0];} //so that there won't be
negative # in log

//linear regression
fladd1 = (firstmin-lastmax+1);
for(i=0;i<fladd1;i++)
{
    lny = log(dataB[(lastmax+i)]);
    sx += dataBtime[(lastmax+i)];
```

```
    sx2 += (dataBtime[(lastmax+i)]*dataBtime[(lastmax+i)]);
    sy += lny;
    sxy += (dataBtime[(lastmax+i)]*lny);
}
sxxsx = sx*sx;
fenmu = ((double)fladd1)*sx2-sxxsx; //means denominator in Chinese
if((fenmu != 0)&&(((fladd1*sxy)-(sx*sy)) != 0)) //get peak and tau
{
    peak = exp((((sy*sx2)-(sx*sxy))/fenmu))+asymp;
    tau = -1/(((fladd1*sxy)-(sx*sy))/fenmu);
}
else {quality = 0;}
```

Variable *firstmin* was the first minimal point in the curve. Now, it represents the end of the region that is used for linear regression (solid section in Fig. 4.5B). It is important to adjust *firstmin* to ensure that all data points in the natural logarithm are positive within the region. Negative points might exist if the trace is noisy (i.e. current crosses the middle line occasionally) or there are unexpected glitches in the trace. The peak current and the steady-state current are obtained after linear regression.

All valid data sets are added and averaged to generate one "digitized" data set (*Ra*, *Rm* and *Cm*) in MATLAB using Eq. 3.5 and Eqs. 3.9-11.

Square-wave perturbation – based on fitting the transferred charges

To increase the signal-to-noise ratio of I-SQA, my mentor suggested me to develop an algorithm using integrated charges (Q-SQA). Since noise tends to cancel each other after summation, the outputs of curve fitting procedures in Q-SQA are of better quality compared with I-SQA. Considering a decaying exponential curve with peak amplitude and asymptote of a and b, respectively, I subtract b from the net current and only integrate the gray area as shown in Fig. 4.6A. The integrated charge value, Qs, is expressed as,

$$Qs = \int_0^t (I - b)\, dt = (a - b)\tau (1 - e^{-t/\tau}) \qquad (4.25).$$

Without subtracting *b* from the current before integration, the trace (Fig. 4.6B) will keep going up without reaching a steady state.

To get the time constant using *Qs*, I use Eq. 3.2 to estimate the asymptote ((*a*-*b*)τ) for *Qs* first, and then invert the trace by subtracting *Qs* from the estimated asymptote (Fig. 4.6C, black trace). The resulting trace is,

$$Y = (a - b)\tau e^{-t/\tau} \qquad (4.26).$$

Again, linear regression of *t* against the natural logarithm of *Y* gives the time constant and (*a*-*b*)τ. Since *b* and τ are known, *a* can be obtained accordingly. To show you that Eq. 3.2 is still valid for a function like Eq. 4.25, consider that three equally-spaced points *A*, *B*, and *C* can be expressed as,

$$A = (a - b)\tau - m \qquad (4.27),$$

$$B = (a - b)\tau - mn \qquad (4.28),$$

$$C = (a - b)\tau - mn^2 \qquad (4.29),$$

where $n = e^{-\Delta/\tau}$, and the solution for (*a*-*b*)τ is still Eq. 3.2.

Also note that *Qs* is only part of the charges that are used to charge the membrane, the total membrane charges are,

$$Q = Qs + \int_0^t 2be^{-t/\tau}\, dt = (a + b)\tau (1 - e^{-t/\tau}) \qquad (4.30).$$

I understand that *Q/V* is *Cm*, however, in order to use the same MATLAB codes for Q-SQA and I-SQA, I decided to output the peak current (*a*) rather than the charges (*Q*) in Q-SQA.

It seems that we have got every equation for Q-SQA, however, it is not true... The clamping time constant in whole-cell recording ranges from tens to hundreds of microseconds. For a data acquisition board acquiring data at 100 kHz, the time interval between two sample points is 10 μs, which is in the same time scale of the clamping time constant. Direct summation of the product of the current and time interval will over estimate the transferred charges in this condition because of the gray areas shown in Fig. 4.7A. To show it mathematically, it is,

$$\sum I_t \Delta \geq \int I_t \, dt \qquad (4.31).$$

According to simulation, unless the Δ is 40 times faster than the clamping time constant, the summation of the product can not approach the actual transferred charges properly (not shown). That is, if τ is 40 μs, the acquisition speed for a single channel has to be at least 1 MHz, and for 3 channels the speed has to be at least 3 MHz,. This kind of board is not available as of March, 2008. The phenomenon is simulated in MATLAB and shown in Fig. 4.7B. The black trace shows the summation, and the gray trace represents the actual transferred charges.

To solve this problem without waiting for new technologies, I derived an equation to correct the estimated *Qs*. Current acquired in an acquisition interval of Δ is shown in Fig. 4.7C. White area represents real transferred charges, which is,

$$m_i \tau (1 - e^{-\Delta/\tau}) \qquad (4.32).$$

The rectangle area used for charge summation is,

$$m_i \Delta \qquad (4.33).$$

Thus, the transferred charges can be corrected by multiplying with the ratio of Eq. 4.32 and Eq. 4.33,

$$Qs' = Qs \cdot \tau (1 - e^{-\Delta/\tau})/\Delta \qquad (4.34).$$

To validate the correction, I use the following MATLAB codes.

MATLAB codes for validating Q correction

```
clear
a = 10; %peak = a+b, unknown
b = 5; %asymptote, known
tau = 100e-6;
interval1 = 10e-6; %DAQ speed.

t1 = (0:interval1:3e-3)';
I1 = (a)*(exp(-t1./tau))+b;
noise = 0*(rand(length(t1),1)-0.5)*3e-1;
I1 = I1+noise;
Q1 = cumsum(I1(1:end-1,1))*interval1; %direct summation
Q1 = cat(1,0,Q1);
QA = (a)*tau*(1-exp(-t1./tau))+b*t1; %theoretical value
Qs1 = Q1-b*(t1); %subtracting DC charges
Qs_corrected = Qs1;

QsA = (a)*tau*(1-exp(-t1./tau)); %theoretical value after subtracting
DC charges
Qs_corrected = Qs_corrected*tau*(1-exp(-interval1/tau))/interval1;
Error = (Qs1(end,1)-QsA(end,1))/QsA(end,1)*1e2
Error_corrected = (Qs_corrected(end,1)-QsA(end,1))/QsA(end,1)*1e2

figure;
plot(t1,Qs1);
hold on;
plot(t1,QsA,'Color','red');
hold off;
figure;
plot(t1,Qs_corrected);
hold on;
plot(t1,QsA,'Color','red');
```

```
hold off;
```

The theoretical trace is in red and the charge summation is in blue. As shown in Fig. 4.8, two traces overlap each other after correction. The errors are ~5% and ~-1.5*10^{-12}% before and after correction, respectively.

I hope I have convinced you that Q-SQA has its strong mathematical supports. However, there is still something in my mind, that is, if τ is obtained before charge correction, and it is then used in charge correction, is τ and the correction still valid? We need some mathematical support to strengthen our belief. The summation of the charges can be written as a geometric series,

$$Qs = \sum_{p=0}^{p=n-1} (a-b)\, e^{-p\Delta/\tau} \cdot \Delta \qquad (4.35).$$

The general solution for a geometric series is,

$$S = \frac{A(1-r^n)}{1-r} \qquad (4.36),$$

where r is the factor, n is the number of members in the series, and A is the first member in the series. So, Eq. 4.35 can be expressed as,

$$Qs = \frac{(a-b)\,\Delta(1-e^{-n\Delta/\tau})}{1-e^{-\Delta/\tau}},\ n \in N \qquad (4.37).$$

By defining

$$A \equiv \frac{(a-b)\,\Delta}{1-e^{-\Delta/\tau}} \qquad (4.38),$$

Eq. 4.37 can be re-written as,

$$Qs = A(1 - e^{-n\Delta/\tau}) \qquad (4.39).$$

Since $n\Delta$ is exactly an expression of time, t, Eq. 4.39 is still a raising exponential function with the same time constant of τ.

Q-SQA – computer implementation

The Q-SQA routine is in a function celled *SqQ* in *CapEngine4*. The implementations of Q-SQA and I-SQA are basically the same except that the integrated charges are used for curve fitting in Q-SQA. Theoretically, the steady-state current can also be calculated from the integrated charges, that is, by estimating the steady-state slope of the charges ($\int Idt$). However, according to my experience, this approach gives noisier outputs compared with values estimated directly from the current. Thus, the way that the steady-state current is estimated is the same in Q-SQA and I-SQA.

If the steady-state current is valid, the fitting routine subtracts it from the total current and then proceeds the integration (Eq. 4.25) using the following codes.

```
C codes for charge integration
dataB[0] = 0;
for(i=1;i<SPC;i++)  //notice that i starts at 1
{
    dataB[i] = dataB[i-1]+((dataA[(indexpeak[np]+i-1)]*signB-
asymp)*interval); //Qs=int((I-asymp)*dt)
}
//estimate peak*tau from Q-subtracted (dataB)
s1=0;s2=0;s3=0;
for(i=0;i<w;i++)
{
    s1 += dataB[(sp1+i)];
    s2 += dataB[(sp2+i)];
    s3 += dataB[(sp3+i)];
}
```

```
s1 /= w;
s2 /= w;
s3 /= w;
if((2*s2-s3-s1) != 0) {PeakTau = (((s2*s2)-(s1*s3))/(2*s2-s3-s1));}
else {quality = 0;}
```

Variable *interval* is calculated at the very beginning using the following code when *CapEngine4* is called.

```
interval = (time[(int)(aiSamplesPerTrigger-1)]-time[0])/
(aiSamplesPerTrigger-1);
```

The value is then passed to *SqQ* with other data and parameters. After summation, the value of steady-state charges ($(a\text{-}b)\tau$) is estimated using Eq. 3.2 and then assigned to variable *PeakTau*.

```
C codes for peak and tau estimation

//Reverse the Q-subtracted curve for fitting
for(i=0;i<SPC;i++) {dataB[i] = PeakTau-dataB[i];}

...adjust lastmax and firstmin etc...

//linear regression of lny v.s. x
...
fenmu = ((double)fladd1)*sx2-sxxsx;
if((fenmu != 0)&&(((fladd1*sxy)-(sx*sy)) != 0)) //get peak and tau
{
    tau = -1/(((fladd1*sxy)-(sx*sy))/fenmu);
    PeakTau *= (tau*(1-exp(-interval/tau))/interval); //correct int(Q)
    peak = PeakTau/tau+asymp;
}
else {quality = 0;}
```

After inverting *Qs* (Eq. 4.26, Fig. 4.6C), the time constant is obtained as described previously. Variable *PeakTau* is then adjusted using the time constant and Eq. 4.34. As in I-SQA, all valid data sets (a, b, τ) are added and averaged to generate one "digitized" data set in MATLAB using Eq. 3.5 and Eqs. 3.9-11.

Square-wave perturbation – validating the algorithms using Simulink

To validate the algorithms, I generate model current first and then check if I-SQA and Q-SQA can retrieve the cell parameters properly from the current. Rather than using an exponential function to generate the signal, I use Simulink, a MATLAB component, to mimic the physical properties of the cell circuits. The model diagram is shown in Fig. 4.9A. Cell parameters can be adjusted in the model (Fig. 4.9A, red circles) in an absolute term. For example, *Ra*, *Rm*, and *Cm* in Fig. 4.9A are 3 MΩ, 50 MΩ, and 20 pF, respectively. The wave protocol (square pulses in this case, Fig. 4.9A, a) drives the circuit, and the total current (b in Fig. 4.9A, black in B), current charging the membrane (c in Fig. 4.9A, red in B), and current leaking through the membrane (d in Fig. 4.9A, blue in B) are displayed. The sampled data (at 100 kHz) are exported (e in Fig. 4.9A) to the Workspace of MATLAB for analysis.

To explain a little bit more about how the model works, let's follow the numbers in Fig. 4.10. Total current flowing through the electrode is driven by voltage difference between the electrode and the membrane (*Ra* is in between). In step 1, membrane potential (from step 5) is subtracted from command potential, and then divided by *Ra* in step 2 to get the actual current flow. Part of transferred charges (step 3) are used to charge the membrane (step 4), and part of them are leaking out of the cell (step 7). Accumulation of the charges on the membrane builds up membrane potential, and the value is *Q/Cm* (step 5). As the membrane potential increases, current leaking out of the cell increases, and the value is *Vm/Rm* (step 6). The leaked charges (step 7) is also subtracted (step 4) from the total charges (step 3). To monitor the net current used to charge the membrane,

the derivative of total membrane charges is calculated in step 8.

The sampled model current is analyzed using the following MATLAB codes.

MATLAB codes for algorithm validation

```
clc
V = 10; %in mV
duration = 0.0025; %half pulse duration. in sec
StdCap = 2; %10pF/1
StdRa = 3;  %1MOhms/1
StdRm = 50; %1MOhms/1
time = simout.time(1000:end);
trigger = simout.signals.values(1000:end,1);
current = simout.signals.values(1000:end,2);
current2 = current+0.03*(rand(length(current),1)-0.5);
[asymp peak tau] = SqCF5(trigger,current,time,3001,3,-5);
[asymp2 peak2 tau2] = SqQ3(trigger,current,time,3001,1,-5);

F = (1-exp(-duration/tau))/(1+exp(-duration/tau));
G = 2*asymp*(asymp*F+peak)/V/(peak-asymp);
Rm = 1./G;
Ra = V/asymp/2-Rm;
Cap = tau.*((1./Ra)+G)*100000;
G = G*1000;

F = (1-exp(-duration/tau2))/(1+exp(-duration/tau2));
G2 = 2*asymp2*(asymp2*F+peak2)/V/(peak2-asymp2);
Rm2 = 1./G2;
Ra2 = V/asymp2/2-Rm2;
Cap2 = tau2.*((1./Ra2)+G2)*100000;
G2 = G2*1000;
% if alpha*beta == 1, 1V = 10pF
% if alpha*beta == 1, 1V = 1MOhms
% if alpha*beta == 1, 1V = 1nS
errCap = (Cap/StdCap-1)*100
errCap2 = (Cap2/StdCap-1)*100
errRm = (Rm/StdRm-1)*100
errRm2 = (Rm2/StdRm-1)*100
errRa = (Ra/StdRa-1)*100
errRa2 = (Ra2/StdRa-1)*100
```

Functions *SqCF5* and *SqQ3* are two separate functions for I-SQA and Q-SQA, respectively. They are created for algorithm development and will be implemented into *CapEngine4* once the routines are validated. In the absence of noise and a filter function, the errors of *Ra*, *Rm*, and *Cm* for both methods in this case are (in %) 10^{-11}, 10^{-10}, and

10^{-10}, respectively.

Square-wave perturbation – PSD analysis of SQA data

As mentioned in the Introduction, PSD always gives better signal-to-noise ratio compared with SQA. Since square waves are composed of sine waves with different harmonies and amplitudes in the Fourier space (Thompson et al., 2001), PSD can also be applied when square pulses are given. However, since the phase angle might be changing during the experiment, it is important to keep in mind that the PSD result might not be valid if the clamping time constant changes a lot. To find the optimal phase angle offline in Capmeter6v3, the program calls a function named *PhaseMatcher2.mexw32*, which is written in C. The *PhaseMatcher2* scan and shift the phase angle from -π to π and check the cross correlation between phase-shifted *Y* (or *X*, or both) from PSD and *C* (or *G* or both) from SQA. The angle that gives highest cross correlation, phase-shifted *X* and *Y* at the angle, and the correlation coefficient are returned from the function. The equation for correlation coefficient is,

$$R = \frac{\sum (Y_i - \bar{Y}) \cdot (C_i - \bar{C})}{\left(\sum (Y_i - \bar{Y})^2\right)^{1/2} \left(\sum (C_i - \bar{C})^2\right)^{1/2}} \qquad (4.40),$$

which can be found at http://local.wasp.uwa.edu.au/~pbourke/other/correlate/index.html. An example is shown in Fig. 4.11. The noise of Q-SQA acquired data (gray) is suppressed after PSD analysis (black). Part of the MATLAB codes handling offline phase shift are shown below.

```
MATLAB codes for offline phase shift
% --- Executes on button press in PhaseShift.
```

```
function PhaseShift_Callback(hObject, eventdata, handles)
XData = get(handles.plot1,'XData');
S = size(XData);
if S(1,2) < 2 %empty plot
    return
end
[index1,index2] = IndexLoc(handles.aitime,XData(1,1),XData(1,end));
XData = handles.aitime(index1:index2,1);
if handles.menuindex(1,1)
    Y = handles.PSDofSQA((index1:index2),1); %Capacitance, at Ch1
    X = handles.PSDofSQA((index1:index2),2); %Conductance, at Ch2
else
    Y = handles.aidata((index1:index2),1); %Capacitance, at Ch1
    X = handles.aidata((index1:index2),2); %Conductance, at Ch2
end
```

The codes here assign data for offline phase shift. If square waves were applied, *X* and *Y* are assigned from *handles.PSDofSQA*, else they are assigned from *handles.aidata* directly.

```
PS = str2double(get(handles.Phase_Shift,'String'));
R_c = []; %R is the correlation coefficient
R_g = []; %R is the correlation coefficient
M = []; %method of correlation
if (handles.menuindex(1,1) && ~isnan(handles.shiftswitch) && (abs(PS-
handles.shiftvalue)) < 0.0001)
    [PS,Cap,Cond,R_c,R_g] =
PhaseMatcher2(handles.shiftswitch,handles.aidata((index1:index2),
1),handles.aidata((index1:index2),2),Y,X,0.1);
    handles.shiftvalue = PS;
    set(handles.Phase_Shift,'String',num2str(PS));
    if (handles.shiftswitch == 1)
        M = 'G';
    elseif (handles.shiftswitch == 0)
        M = 'C';
    else
        M = 'C+G';
    end
    disp(['CorrMethod:',M,' P:',num2str(PS),' R_c:',num2str(R_c),'
R_g:',num2str(R_g)]);
```

The function calls *PhaseMatcher2* to scan the phase only when all three criteria are met: 1. SQA were used, 2. *handles.shiftswitch* is not a NaN, and 3. the input value in the 'Shift' edit box is not modified. Variable *handles.shiftswitch* is a NaN when the 'User-defined' option in the 'Shift' context menu is selected. If the 'Shift' edit box is modified,

the program will use the modified value to adjust the angle rather than calling *PhaseMatcher2* to scan the phase. The correlation method, phase angle, and the correlation coefficient at the angle are displayed in the MATLAB Workspace.

```
else
    %disp('entered');
    handles.shiftvalue = PS;
    PS = PS*pi/180;
    Cap = (-X*sin(PS))+(Y*cos(PS));
    Cond = (X*cos(PS))+(Y*sin(PS));
    if handles.menuindex(1,1)
        R_c = PhaseMatcher2(handles.aidata((index1:index2),1),Cap);
        R_g = PhaseMatcher2(handles.aidata((index1:index2),2),Cond);
        disp(['CorrMethod:N/A P:',num2str(handles.shiftvalue),'
R_c:',num2str(R_c),' R_g:',num2str(R_g)]);
    end
end
guidata(hObject,handles);
```

If the user modified the value in the 'Shift' edit box, the function adjust the angle as specified. Function *PhaseMatcher2* is called to calculate the correlation coefficients for *C* and *G* at the specified angle if square pulses were used.

Notice that for cross correlation between *X* and conductance, I did not take *Ra* into account. I understand that the overall conductance is determined by both *Ra* and *Rm*, however, if *Ra* fluctuates a lot during the experiment, the phase angle must also varies a lot. In this case, it does not make sense to find a “fixed” phase angle to fit the overall conductance trace, so I use only *Rm* for correlation. Again, please keep in mind that the PSD analysis may not be valid if the clamping time constant changes a lot during the experiment.

4.3 Capmeter 6

What is it?

Capmeter 6 is a tool for measuring membrane capacitance. It can use either sine waves or square waves to estimate cell parameters. Phase-sensitive detection (PSD) is utilized in both cases. If square wave perturbation is selected, one can chose to use either integrated charges (Q-SQA) or direct current trace (I-SQA) for calculating the cell parameters. Other functions like digital filtering, pulse stimulation, offline phase angle adjustment, baseline subtraction, and data normalization are also implemented.

System requirements

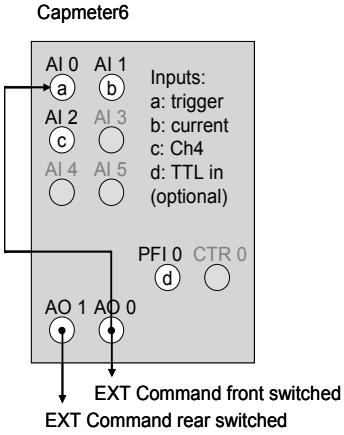
Using MATLAB R2006b or R2007a is recommended. Versions earlier than R2006b are not supported. R2007b is also not recommended because the 'Foregroundcolor' and 'Backgroundcolor' properties of GUI buttons do not work properly. However, if you need to synchronize the recording with an external TTL signal (e.g. image acquisition software), you need to use R2007b because the 'HwDigital' trigger of AO object is not functioning in earlier versions.....

It is reported by my colleague that Capmeter is not compatible with MATLAB R2008a... It is always quite painful to deal with this kind of compatibility issues especially when the program is quite complex. The even worse thing is that we do not have R2008a license for the lab. So, you may download R2007 from the Mathworks to run Capmeter if you have the license, and there is no solution for the compatibility issues temporarily as of March 2008.

The program is developed for DAQ boards from National Instruments. However, you can use other DAQ boards supported by the MATLAB Data Acquisition Toolbox. Some adjustments of the codes may be required. The minimal AI and AO speed for the default setting is 300 kHz and 200 kHz, respectively. I use PCI-6052E from NI (AI 333 kHz, AO 333 kHz), but I recommend using the M-series board like PCI-6251 (AI 1.25 MHz, AO 2.8 MHz) since it is cheaper and faster.

The program works fine with Pentium4 2.53 GHz, 1GB RAM, 25 GB hard drive free space, and Window XP SP2.

Connection diagram



Analog output AO0 is used to send out either sine or square pulses. It is also connected to AI0 to trigger the acquisition. Output channel AO1 is used to generate stimulating pulses when the 'Pulse' button is clicked. After connecting AO channels to external command inputs of the patch clamp, be sure to check the command sensitivities in the M-file. The default values for AO0 and AO1 are 20 mV/V and 100 mV/V, respectively. You may edit the values of *handles.aoCh1convert* (for AO0) and *handles.aoCh2convert* (for

AO1) in the CapmeterSetting.m file, or change the default values directly in ~line 155 of Capmeter6v3.m.

Input AI1 receives current from the patch clamp. The acquired data are used for estimating cell parameters. I recommend you to low-pass filter the input at 10~20 kHz. Although all the algorithms works fine even when the hardware filter is bypassed, using a 10~20 kHz filter will reduce the noise greatly. Low-pass filtering the current at 2 kHz or slower is also not recommended especially when I-SQA is selected, because the shape of the peak current is severely affected in this case. Channel AI2 records whatever is connected to it. The digitally filtered data is displayed at channel 4.

It is possible to synchronize the acquisition of Capmeter with another program. In this case, a TTL input to PFI0 is required. Capmeter does not support TTL output simply because the MATLAB does not support buffered digital output.

Things you need to know before using

1. Please cite the paper: Wang and Hilgemann, 2008.
2. The 'digital filter' in Capmeter only calculates the average of data acquired in a specified time window. Although the program automatically adjusts the time window so that the command sine or square waves are canceled, it is still possible that the 'filtered' current does not represent the actual current faithfully when data clipping happens.
3. When PSD is selected, uncheck the 'Sine' check box results in the generation of triangular waves. When SQA is selected, it does not matter if the check box is

checked or not.

4. For PSD, the phase angle has to be adjusted again whenever you start a new acquisition even for the same cell. It is also recommended to check the phase angle whenever the AO system is restarted (i.e. changing oscillation frequency or amplitude, giving pulses).

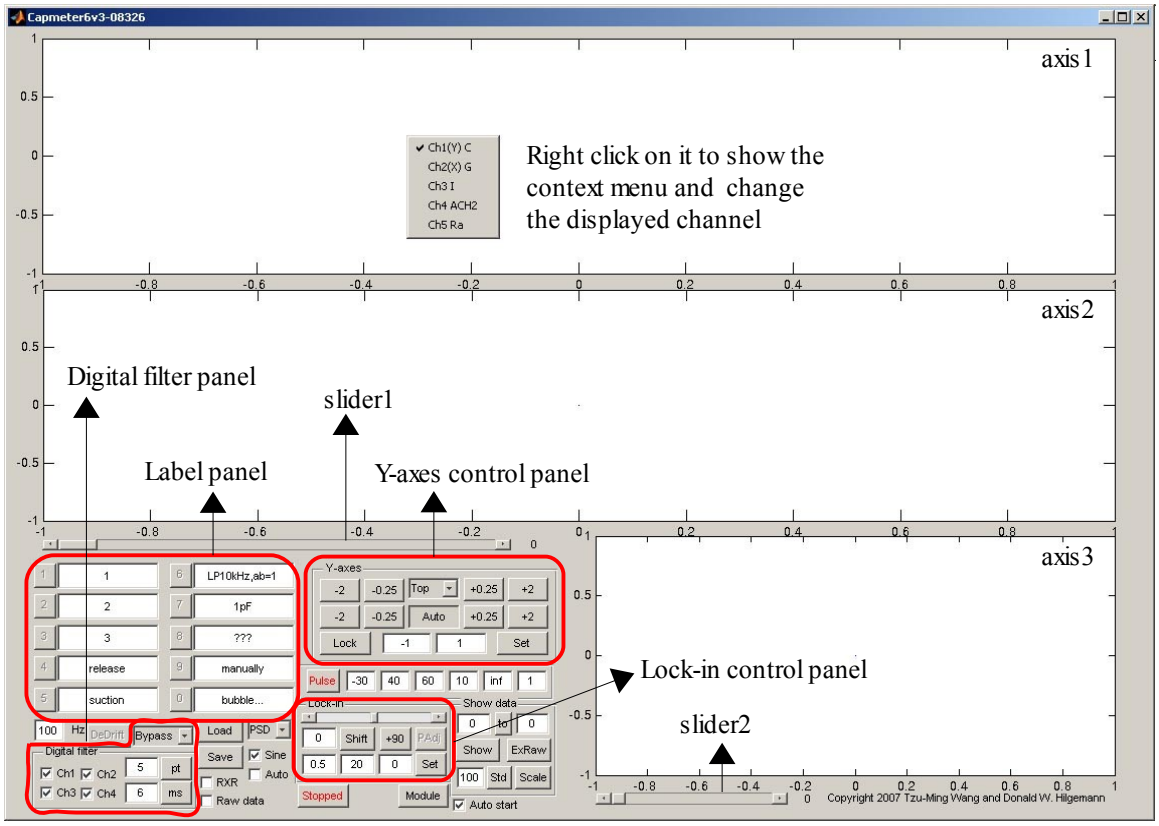
5. In the presence of noise, I-SQA gives better (closer to the actual value) estimation of membrane capacitance compared with Q-SQA. However, unlike Q-SQA, which is almost unaffected by the filter function, the estimation of I-SQA is greatly influenced by the filter.

6. Q-SQA is the preferred SQA method because the estimation of *Cm* is not affected by *Ra* fluctuation according to my experience.

Running the program

In MATLAB, change the 'Current Directory' to the one containing Capmeter components and then right click on Capmeter6v3.m file (double click on the file will open the M-file editor). Click 'Run' in the context menu and MATLAB will launch the

program as shown below.



Before starting the acquisition, you might want to adjust the sampling frequency (points per second to be digitized from the raw data). The default setting is 100 Hz, you can modify the setting in CapmeterSetting.m to change the start-up value. After selecting the desired algorithms (PSD, I-SQA, Q-SQA) from the pop-up menu, you may adjust parameters (frequency, amplitude) for command potential generation in the 'Lock-in control panel', and then start the recording by clicking the red 'Stopped' button. In case if you forget the definitions for the edit boxes, you can always move the mouse cursor onto the box for few seconds and the description will pop out.

The scale of the Y axes is controlled by 'Y-axes control panel'. To change the setting

for axis1, you first select 'Top' ('Middle' for axis2, 'Bottom' for axis3) from the pop-up menu, and then adjust the values. The first row is for the upper limit and second row is for the lower limit. You may also specify the settings by putting values into the edit boxes and then press 'Enter' (pressing 'Set' is not necessary). MATLAB sets the limits automatically if the 'Auto' button is pressed. By pressing the 'Lock' button, the scale of axis2 is locked to axis1. It is useful to use 'Lock' when PSD is selected, because one can check the phase angle by adjusting capacitance compensation and then compare the relative changes in channel 1 (*Y*) and channel 2 (*X*). Also notice that if all of the data are out of the range of axis1 settings, the X axes of axis1 and axis2 become 0 to 1 without updating. To correct the situation, you can either adjust the Y-axes settings for axis1 or simply press the 'Auto' button to reset axis1. Similar effect also happens to axis3, and the solution for it is the same.

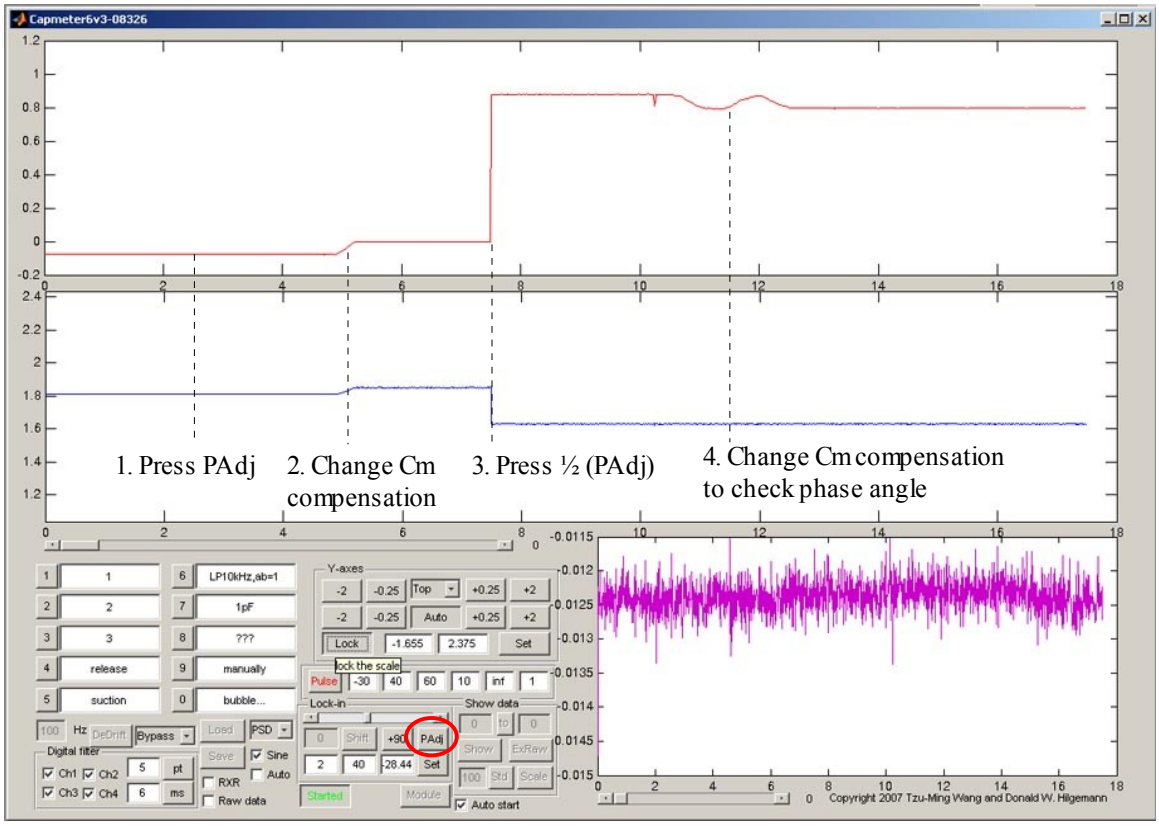
The X axes are controlled differently online and offline by two sliders. Slider1 controls axis1 and axis2, and slider2 controls only axis3. In the online mode, the values (let's say *n* for example) of the sliders indicate that the last *n* seconds of data are displayed. If the value is zero, all of the data are shown. The default maximal values for slider1 and slider2 are 120 and 50 seconds, respectively. You may change it from CapmeterSetting.m. In the offline mode, the sliders represent the entire data set. You may “navigate through” the entire data set by sliding the slider, and the length of the data shown on the screen is defined in 'Show data panel', which will be introduced in later section.

To make notes, you can use the 'Label panel'. Buttons 1~5 put notes on axis1, and

buttons 6~0 put notes on axis2. You may also put notes by pressing the numbers on the keyboard during the recording. Other Keypress functions are, '*' calls the 'Module' if the program is stopped, and 't' puts time constant on axis1 when SQA is selected.

The 'Module' button by default calls Capmodule4.m, which is a tool for making seals. If the 'Auto start' check box is checked, Capmeter starts automatically when the module is closed.

Using PSD



If PSD is selected, the first thing you need to do after starting the recording is to adjust the phase angle. Capmeter can calculate the phase angle for you when you use the 'PAdj' (phase adjustment, in red circle) button in the 'Lock-in control panel' as shown

above. You first click the button (PAdj becomes ½, meaning step 1 of 2), and the program takes the averages of 20 data points from channels 1 (red) and 2 (blue). After changing the compensated capacitance (red), click on '½' (PAdj) button again, and Capmeter get the averages and then adjust the phase angle automatically using Eq. 4.6. When the phase angle is properly adjusted, changing capacitance compensation does not affect the reading in conductance channel (blue). You may also set the angle by putting specific value in the edit box and then press 'Enter', or play with the slider bar in the panel. Occasionally, you might get an angle that is ½ π or π (or so on) away from the actual angle when using 'PAdj'. In this case, it is also useful to use the '+90' button to correct the angle.

To adjust the phase angle offline, simply put a value in the edit box next to the 'Shift' button, and then click 'Shift'. For PSD analysis of SQA data, you can right click on the 'Shift' button and select a method for cross correlation from the context menu, and then click 'Shift' to start phase scanning. Detail about PSD analysis of SQA data can be found in the last paragraph of section 4.2.

Besides adjusting phase angle online, you can also change oscillating frequency and the peak-to-peak amplitude during the recording. However, since Capmeter uses trigger signal added on top of the command oscillation to synchronize data acquisition, the new amplitude must be smaller than the trigger signal. By default, you may not adjust the amplitude 1.2 mV (if command sensitivity is 20 mV/V) above the original setting. You can lower the amplitude to whatever value you want, but remember, there is still a trigger signal that is 1 mV larger than the initial peak amplitude (not peak-to-peak).

Using SQA

Although the coding for SQA is quite difficult and has spent me a huge amount of time, using SQA is quite easy. The first step is to select the algorithm, and the second step is to click the 'Stopped' button to start recording. That's it. If the net gain ($\alpha\beta$) is 1, then 10 pF/1 for *Cm*, 1 nS for *Gm*, and 1 MΩ for *Ra*. The hints will show up when you move the mouse cursor onto the algorithm pop-up menu for few seconds.

Note that if the SQA letters become red (*Foregroundcolor*), that means the fitting routines fail to extract the information, and NaN is assigned to all cell parameters. If the entire pop-up menu become red (*Backgroundcolor*), that means the peak current might have overload the board (>10 V), and the command amplitude should be decreased. I will recommend you to restart the recording in this circumstance if possible.

For your information, there is an 'Auto' check box beneath the 'Sine' check box. It is checked automatically when you select SQA. Right-click on the check box, and the context menu shows you two options: 1. frequency and 2. fitting range. The 'fitting range' option is selected by default, that means the fitting routine uses a 'tau factor' to select a region for fitting (else the entire data range is used). The 'frequency' option is designed to adjust the oscillating frequency and amplitude automatically according to the estimated time constant and peak current, so that the cell membrane can be charged properly, and the signal is kept in an optimal range. However, this function (in Capmeter function *SqAlgo*) is never perfect under my criteria, and I will probably remove it in the future versions. It is recommended to keep the 'Auto' check box checked, and do not change the default settings.

Using digital filters

There are two digital filters implemented. One is the running filter, the other is the background averaging filter. The running filter does not change the acquired data, it simply processes digitized data points that have been shown on the screen. Whenever the axes are refreshed, all the points are processed again by the filter if the filter mode is not 'Bypass'. Two types of running filters are available, one is running mean, the other is running median. Running median filter is good for removing unexpected glitches in the trace. You may select one of them from the pop-up menu in the 'Digital filter panel'. To set a window for calculating mean/median, you put a number in the edit box next to the 'pt' (points) button and then press 'Enter'. Running filtering can be applied on all channels, and it is performed by *Dfilter2.mexw32*, which will be discussed in later section.

The background averaging filter is designated to process signals received from the data acquisition board (i.e. channels AI1 and AI2). After averaging, the digitized data from AI1 (current) are recorded in channel 3, and data from AI2 (whatever is connected) are recorded in channel 4 (channels 1, 2, and 5 are not affected by this filter). You may set the time window for averaging by putting specified value (in ms) in the edit box next to the 'ms' button, and then press 'Enter'. The program will adjust the value automatically so that current resulting from the command oscillation can be cancelled. The filter is implemented in *CapEngine4*, with function name called *Dfilter*.

Variables in the Workspace

After the recording is stopped, Capmeter assigns a number of variables to the MATLAB Workspace. They are introduced below.

DAQinfo, structure containing hardware settings

aiSR, analogue input speed, in Hz

aoSR, analogue output speed, in Hz

aoCh1convert, command sensitivity for AO0, in mV/V

aoCh2convert, command sensitivity for AO1, in mV/V

starttime, initial trigger time of analogue input, in [year month date hr min s]

FigureData, it contains data in axis1 and axis2 when 'Show' button is clicked

[time (s), data from axis1, data from axis2]

PSDlog, cell array containing oscillation parameters

{time, frequency (kHz), amplitude (mV), phase angle (°), wave form-algorithm}

PSDofSQA, it contains PSD analyzed SQA data, [Ch1 (*Y*), Ch2 (*X*)]

Pulselog, structure containing pulse protocol and other information

wavept, number of AO points for each trigger (`length(handles.aodata(:,1))`)

data, structure array containing information for each pulse train

pulse, shape of the pulse protocol (V). NaN is assigned if number of pulses is set to *inf*.

Pulseinfo, [initial step (V), length (AO points), interval (AO points), step increment (V), number of pulses, rounds]

note, reserved for future application

trigger, trigger time, in seconds

data, array containing processed data, [Ch1, Ch2, Ch3, Ch4, Ch5]

labels, cell array containing labels and other information (bin at 0.5 s)

{channel applied, time (s), value of each channel [Ch1~Ch5], label}

rawdata, generated when 'ExRaw' button is clicked (DISABLED by default)

[time (s), AI0, AI1, AI2]

rxr, if the 'RXR' check box is checked, it contains sampled raw data

[time (s), AI0, AI1, AI2]

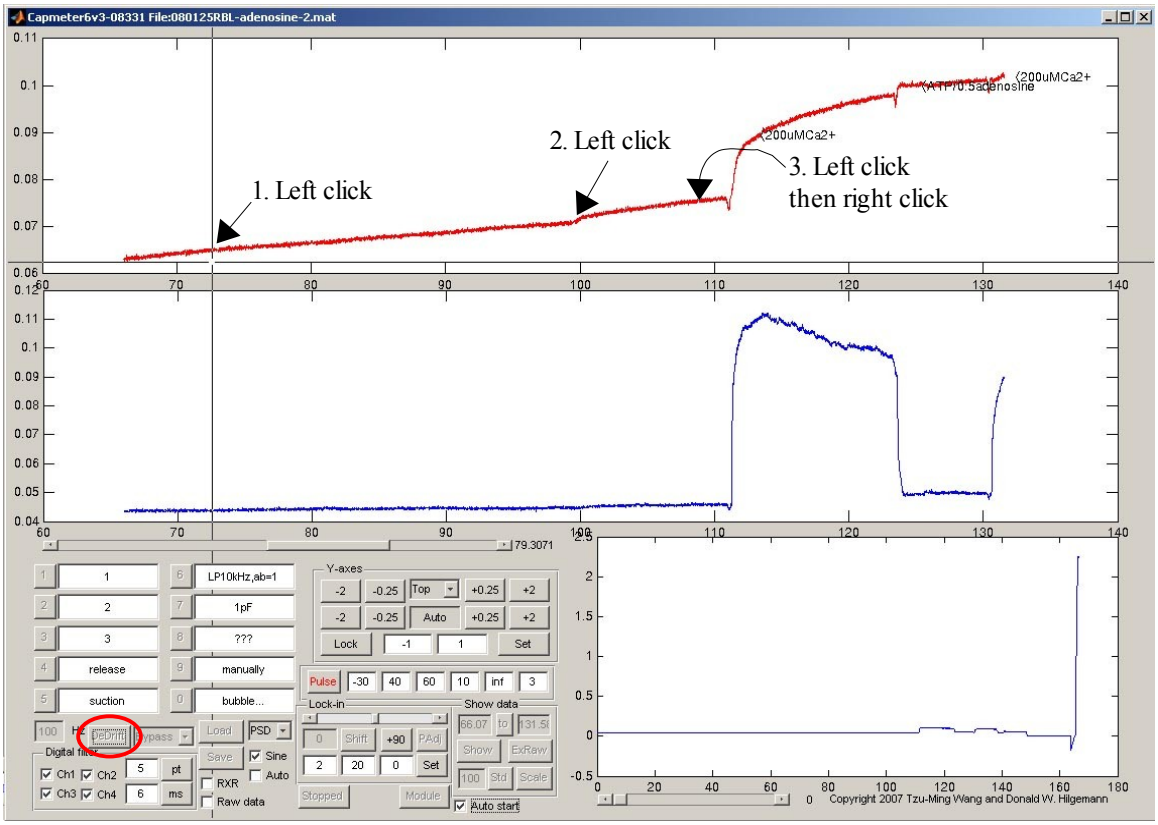
time, array containing time for processed data (in s)

version, structure containing version information

Shell, version of the GUI

Engine, version of the *CapEngine*

Subtracting the baseline

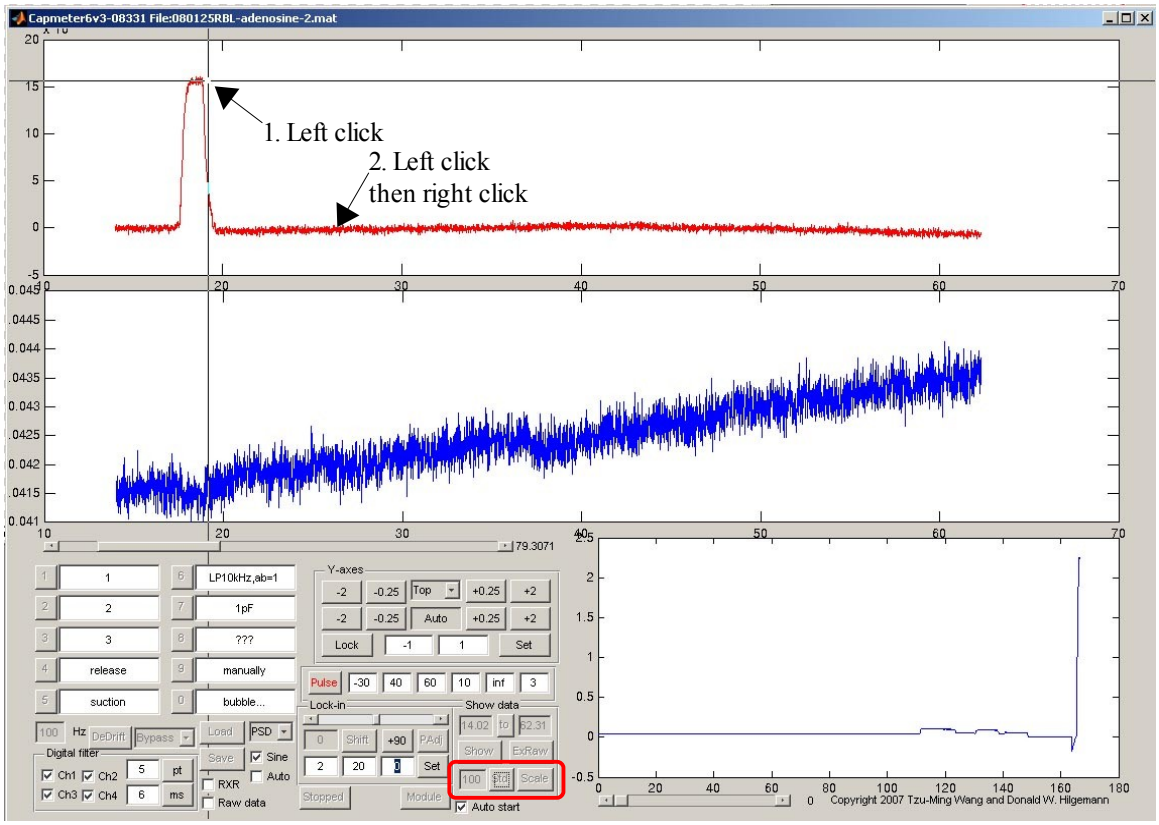


When the solution flow is turned on, solution level in the recording chamber increases, and the increased stray capacitance will be detected if higher oscillation frequency is used (e.g. for excised patch). In this case, subtraction of the increasing baseline capacitance is desired. To do so, you can use the 'DeDrift' button (in red circle). After clicking the button, you use the mouse cursor to select multiple points from the axis to define sections for linear regression, and then right-click the mouse to subtract the baseline using section-specific slopes. For example, baseline between points 1 and 2, and baseline on the left of point 1, are subtracted using slope of points 1 to 2; baseline between points 2 and 3, and baseline on the right of point 3, are subtracted using slope of points 2 to 3. You may select as many points as you want, but make sure that you use this function only when

you are certain about the origin of the drift.

If the first click is not on axis2, data in axis1 will be processed. Also, you don't have to point to a real data point on the trace, because the function only gets the X coordinates from your clicks, Y coordinates are not used in the function at all.

Scaling the data



PSD only gives relative, but not absolute capacitance reading, so I usually use the capacitance compensation knob to make a “standard” peak during the recording, and then convert the unit from V to fF afterward using the “standard”. For instance, if the standard peak represents 100 fF, I put '100' into the edit box next to the 'Std' button (in red circle) and then click the 'Std' button (it might be useful to subtract the baseline using 'DeDrift'

first). After defining the standard using the mouse cursor (2 points), a conversion factor is calculated and stored in the memory. Whenever you want to scale the data on the screen, you just click 'Scale' next to the 'Std' button, and the previously calculated factor will be used to scale the data.

Note that the 'Scale' function only works for Ch1 now, and the first point is defined as zero. Also, if you click the 'Scale' button twice, the data will be scaled twice, too, and of course, it is wrong. When defining the standard, the function only gets the Y coordinates from your clicks, so the X coordinates are not important at all. It is particularly useful to do so when the trace is noisy, because you may define precise Y coordinates by putting the mouse in the middle of the noisy trace (like doing low-pass filtering by eye).

Showing the data

You can use the 'to' button in the 'Show data panel' or the edit boxes next to it to specify a region to be displayed. These two methods are slightly different. When the 'to' button is used, it only crops the data which have been displayed on the screen; when you put values into the edit boxes and press 'Enter', the program retrieves “original data” within the specified time frame and then update the displayed data. It will make a difference after you use the 'DeDrift' function. For example, you may want to subtract the baseline first and then enlarge a portion of the baseline-subtracted data. In this case, you need to use the 'to' button but not the edit boxes.

After selecting the desired region, you may drag slider1 to move along the trace as introduced previously. Notice that slider1 also retrieves original data for displaying. To

show the entire trace again, you need to put '0' (zero) into the edit box on the right of the 'to' button and then press 'Enter'.

By clicking the 'Show' button in the 'Show data panel', a MATLAB figure is generated. The figure contains only traces displayed in axis1 and axis2, and the axes settings and color settings are the same with axes 1 and 2. If you would like to show only 1 channel in the figure, the fastest way is to select the same displayed channel for both axes and then click 'Show'. Variable *FigureData* is assigned to the Workspace, and the format is [time (s), data from axis1, data from axis2]. Note that values in *FigureData* are exactly the displayed data. If you use 'DeDrift' and/or running filter, baseline-subtracted and/or filtered data are exported.

Exporting the data

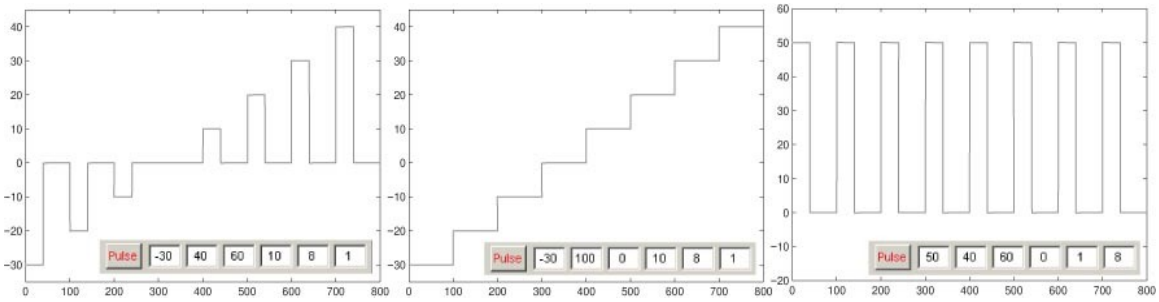
Using the 'Show' button is the only way to export processed data to the MATLAB Workspace as introduced in the previous paragraph.

To export raw data within the specified time frame, you may use the 'ExRaw' button. This function is disabled by default because the MATLAB function *daqread* is EXTREMELY slow when there are lots of NaN in the file. You may enable it by editing the *ExRaw_Callback* in Capmeter6v3.m file, and you will see tons of warnings about NaN when you click it. The exported raw data is in variable *rawdata*. Be sure to check the 'Raw data' check box under the 'Save' button before saving the file if you want to retrieve raw data later. The file (.daq) will be very large, and probably useless... I recommend you not to use it unless necessary.

An alternative way to save and export raw data is to check the 'RXR' (run-time exporting raw data) check box during the experiment. By checking the check box, the program saves raw data for only two consecutive square pulses (or 2 sine waves for PSD) in every processing cycle (0.5 s), and the "sampled" raw data will be assigned to the Workspace in variable *rxr* after recording. You can check and uncheck the 'RXR' check box during the recording, and only the desired sections of raw data are stored. The drawback is that you need to locate the section of interests by yourself after the recording. It is usually not too difficult, but you can always automate the processes by making a M-file for these routine works. You know, my major is neuroscience, not computer science, and my goal should be advancing the science, but not developing a program... So, please understand that I only develop functions that will speed up my research and data analysis.

Giving pulses

You may need to trigger membrane fusion using voltage pulses for certain cell types, or you may want to probe the current-voltage relationship using increasing voltage steps. In these cases, you can use the 'Pulse panel', which is above the 'Lock-in control panel'. The definition for each edit box within the panel will show up when you put the mouse cursor onto the box for few seconds. Three examples are shown below.



The voltage is set to zero during the interval (left panel). You may set 'interval' to zero to make the voltage steps “continuous” (middle panel). If you want stimulating pulses with the same amplitude, you can set 'step increment' to zero and then specify the number of desired pulses (right panel). In this kind of pulse protocol, I recommend you to specify the number of pulses in the 'rounds of pulses' edit box but not in the 'number of pulses' edit box (right panel). Because calculation of the pulse protocol takes time, putting the number in the 'rounds of pulses' edit box avoids redundant calculation for the same pulse and reduces the delay between pushing the 'Pulse' button and the actual pulse generation. If you put *inf* in the 'number of pulses' edit box, the value in the 'step increment' edit box will be neglected.

Although you may get the I-V curve by using the 'Pulse panel', another program called IQplot might fit your application better. IQplot is developed to fulfill my mentor's will, and it will be introduced in section 4.5.

TTL triggering

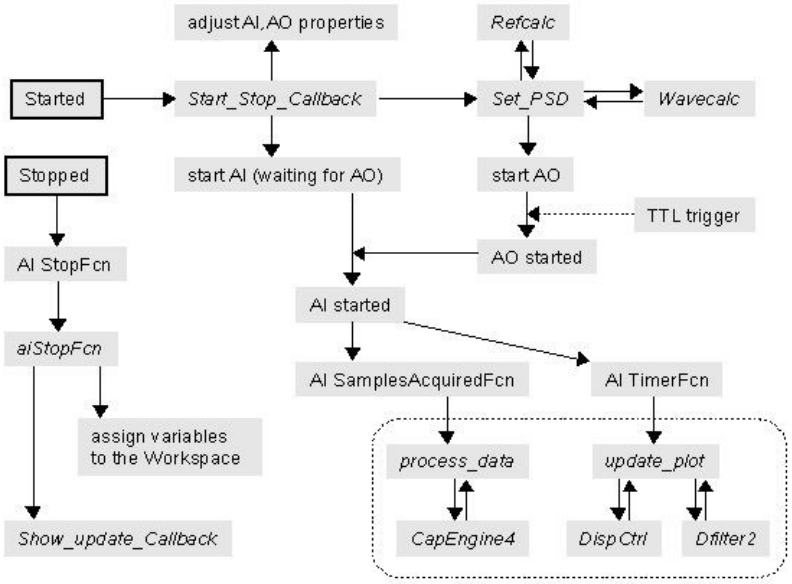
In case if you need to synchronize the acquisition of Capmeter with other programs (e.g. image acquisition software), you may use the TTL triggering function. Connect the external TTL source to PFI0 on the terminal block, and then right click on the 'Stopped' button when Capmeter is stopped. Check the TTL option in the context menu, and then click the 'Stopped' button to start the program. Capmeter is then waiting for the external TTL signal to trigger the acquisition. Once the acquisition is started, Capmeter does not need any other TTL trigger because the acquisition is continuous. Capmeter does not send out TTL signal simply because MATLAB does not support buffered digital output.

Reader mode

You may browse and analyze your data at home without installing a data acquisition board. If the program can not detect the board or there is something wrong with the hardware settings, it will enter the 'reader mode'.

If you are trying to modify the program, be sure not to use the function *isrunning*. Although the MATLAB recommend the users not to access the *running* property of AI and AO objects directly, Capmeter pretends that there is a variable called *running* in the reader mode even when AI and AO objects do not exist. If function *isrunning* is used, errors may occur in other functions in Capmeter6v3 when reader mode is launched. I did not use *isrunning* at the very beginning of the development simply because there was not such a function at that time.

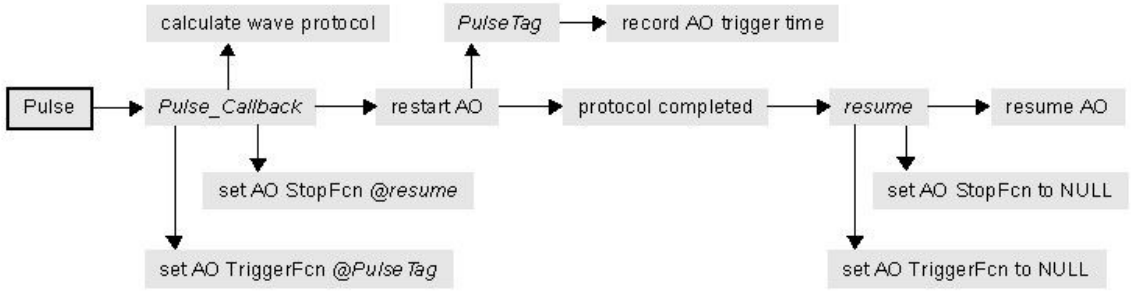
The information flow



This section and the following one introduce how Capmeter6v3 works. The above

diagram shows you the information flow in the program. Function names are in italic, and functions that are evoked periodically during the recording are in the dashed box. When you start the recording, function *Start_Stop_Callback* is evoked. It resets some variables and adjusts properties of GUI components, AI and AO objects etc. Then, function *Set_PSD* is called. This function further calls *Wavecalc* to calculate the oscillation wave form, and calls *Refcalc* to generate reference waves for PSD. AO is started after the output waves are queued into the memory. Note that AO is started in function *Set_PSD* but not in function *Start_Stop_Callback*. If TTL trigger is selected, AO will not start sending out signals unless a TTL trigger is detected.

AI is started in *Start_Stop_Callback*, however, the acquisition will not start until it receives trigger signals from AO. Once the AI is triggered, its *SamplesAcquiredFcn* and *TimerFcn* will be evoked periodically to process the data and update the plots. When the recording is stopped, the AI *StopFcn* will assign the data to MATLAB Workspace, and shows all the data by calling *Show_update_Callback*.



When the 'Pulse' button is pressed, the function *Pulse_Callback* calculates the wave protocol and sets AO *StopFcn* and *TriggerFcn*. Function *PulseTag* is evoked when AO is triggered, and records time point at which the pulse protocol is given. When the protocol

is completed, function *resume* resets the AO properties and then restart the AO.

Main variables in the program

There are tons of variables in the program, and most of them are categorized into groups and have clear descriptions in the codes. A few important variables are introduced below.

handles.bufdir, it contains temporary directory used to save CapBuffer.daq file. If the 'Raw data' check box is checked, CapBuffer.daq will be renamed to FILENAME.daq and moved to the directory indicated by *handles.current_folder*.

handles.current_folder, it is the last folder you visited when you use the 'Save' or 'Load' button.

handles.nidaqid, device ID for the data acquisition board. There might be more than one device ID for the installed board (because of different drivers used). The device ID can be '1', '2', 'Dev1', or 'Dev2' etc. for boards from National Instruments (NI). It is critical to select a proper ID especially for the M-series boards from NI because they usually have more than one device ID. In CapmeterSetting.m file, you may specify the N^{th} device ID you want to use by assigning *handles.nidaqid* = 1 or 2 or etc. For example, if *handles.nidaqid* = 1, and the first item in the *InstalledBoardIds* property of the board is 'Dev2', 'Dev2' is the ID to be used. The value of *handles.nidaqid* will be pointed to the actual device ID later using code `handles.nidaqid = Daqinfo.InstalledBoardIds{1,handles.nidaqid}`.

handles.dispindex, display index used to indicate current channels displayed on the

panels. If *handles.dispindex* = [1,5,2], that means the top, middle, and bottom panels display Ch1, Ch5, and Ch2, respectively.

handles.slider1range, slider1 range in online mode. Default value is 120, that means 2 min.

handles.slider2range, slider2 range in online mode. Default value is 50, that means 50 s.

handles.shiftswitch, method for cross correlation used in 'PSD analysis of SQA data'. 0: *C* correlation, 1: *G* correlation, -1: both *G* and *C* for cross correlation.

handles.aoCh1convert, command sensitivity for AO0, in mV/V.

handles.aoCh2convert, command sensitivity for AO1, in mV/V.

handles.aiSR, acquisition speed for AI, in Hz.

handles.aoSR, acquisition speed for AO, in Hz.

handles.rSR, frequency for digitizing data, in Hz.

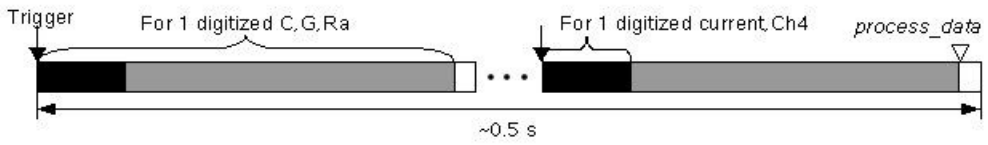
All of the above variables can be adjusted in CapmeterSetting.m file. If Capmeter6v3 can not find the file, default values defined in the main program will be used.

handles.aiSamplesPerTrigger, it is used to set the AI *SamplesPerTrigger* property and calculated using code `floor(((1/handles.rSR)-0.001)*handles.aiSR)`. For instence, if *handles.aiSR* is 100 kHz and *handles.rSR* is 100 Hz (i.e. 10 ms interval), the program will acquire 9 ms of data (i.e. 900 samples) and then wait for the next trigger, which is 10 ms apart from each other.

handles.SpmCount, it is used to set the AI *SamplesAcquiredFcnCount* property. The value

is `(handles.aiSamplesPerTrigger)*round(handles.rSR*0.5)`. With the same example shown above, the program evokes function *process_data* every 0.5 s (i.e. 45000 samples) to extract and process acquired signals.

handles.filterv2p, number of points to be averaged in the background averaging filter. The relationships among *handles.aiSamplesPerTrigger* (black+gray), *handles.SpmCount* (black+gray of the entire time frame), and *handles.filterv2p* (black) are shown below.



handles.aidata, n-by-5 matrix containing all the processed data.

handles.aodata, n-by-2 matrix containing output signals for AO0 and AO1.

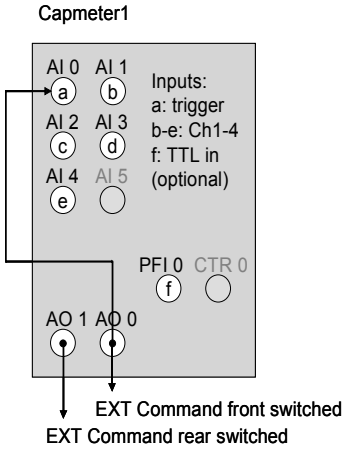
handles.aitime, column array containing corresponding time points for *handles.aidata*.

4.4 Capmeter 1

What is it?

Capmeter 1 serves as a plain chart recorder with digital filters and some other functions. It is modified from Capmeter 6, and inherits most of the functions from Capmeter 6 except the ability to extract cell parameters. You may record at most 4 channels at the same time using Capmeter 1.

Connection diagram



Although Capmeter 1 does not send out sine or square waves, it still uses trigger signals from AO0 to trigger acquisition so that signals can be extracted and processed during recording using MATLAB function *getdata*, please refer to Introduction for detail.

Things you need to know before using

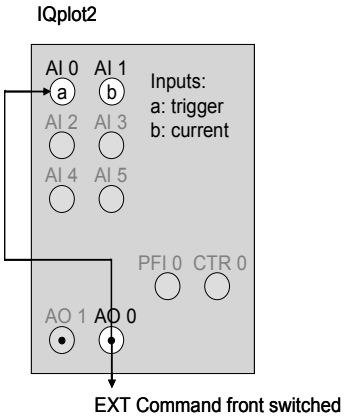
1. Please cite the paper: Wang and Hilgemann, 2008.
2. Please remember to adjust the command sensitivities.
3. Please refer to section 4.3 Capmeter 6 for detailed description and discussion.

4.5 IQplot

What is it?

IQplot is a program to probe current-voltage (I-V) and transferred charge-voltage (Q-V) relationships in voltage clamp configuration. It is developed to fulfill my mentor's will. I don't have any idea about the differences between IQplot and other commercially available software, because I have never used any of them (please refer to the Introduction if you are interested in the history).

Connection diagram



The connection is compatible with that of Capmeters.

Things you need to know before using

1. There is no paper for this program, yet, as of March 2008. Please cite the link.
2. This program is still under development, and there is only limited error check and control.
3. Please remember to adjust the command sensitivity in ~line 144 (default 20

mV/V).

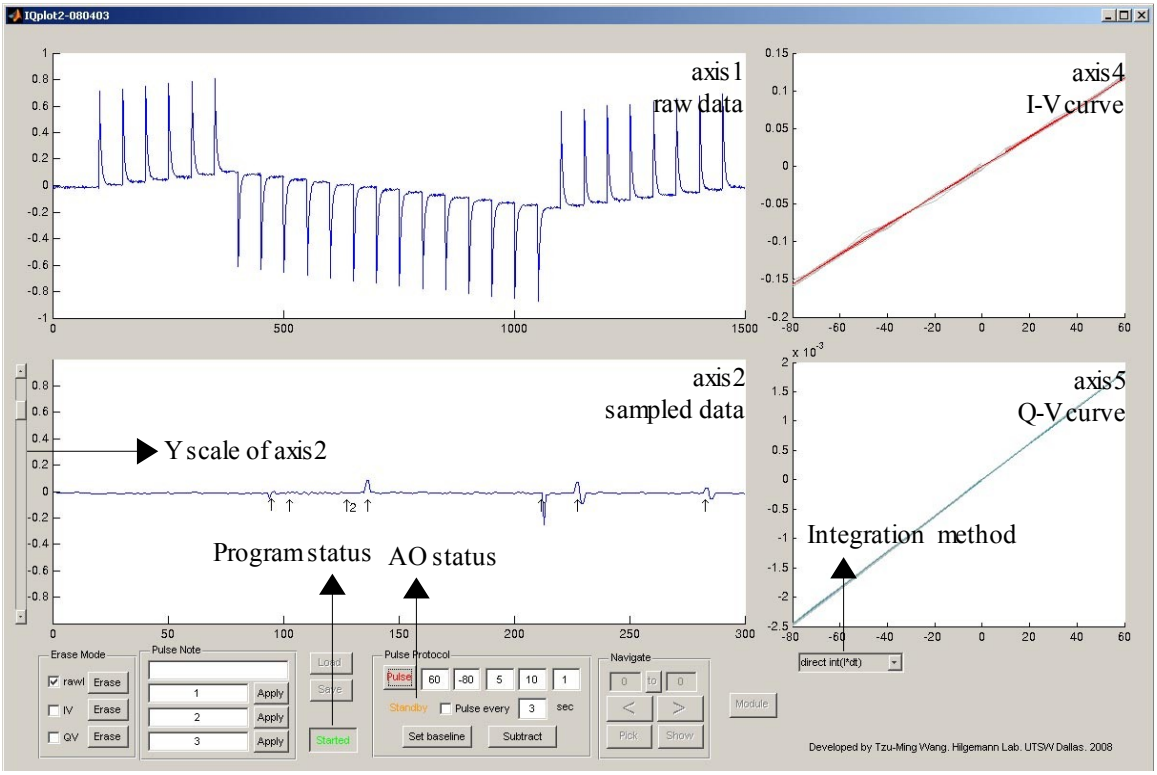
4. Current in the I-V plot is estimated using Eq. 3.2, which is derived from an exponential function, so please do not compensate membrane capacitance completely.

5. There are two methods for charge integration, and you may switch from one to the other during the experiment. Please make a note to specify the method used.

6. Unlike charge integration in Capmeter6v3, IQplot2 does not correct the integrated charges using Eq. 4.34. It is because time constant from a single curve is not accurate and introduces excessive noise if Eq. 4.34 is used.

7. Variable *datasample* in the Workspace does not represent accurate time-signal relationship.

Running the program

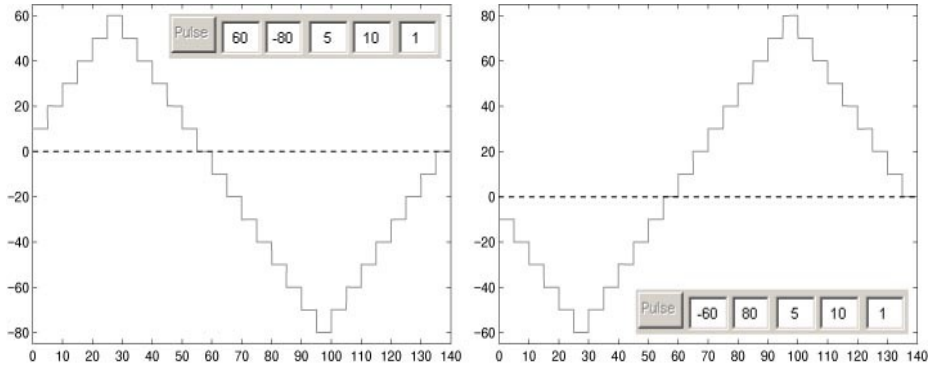


In MATLAB, change the 'Current Directory' to the one containing IQplot components and then right click on IQplot2.m file (double click on the file will open the M-file editor). Click 'Run' in the context menu and MATLAB will launch the program.

Press the 'Stopped' button to start the AI object, and the sampled current is shown on axis2. To show the sampled current, the function *peekdata* is evoked every 0.05 s (i.e. 20 Hz). The peeked data are displayed on axis2, and the last peeked data is stored in *handles.datasample*. Since *peekdata* function does not extract the data from the MATLAB Engine and the returned data may be missed or repeated (please refer to the function references of MATLAB), *handles.datasample* does not represent accurate time-signal relationship. Moreover, if errors occur when *peekdata* is called (e.g. program is

busy) , the program assigns previous value to the current time point. In online mode, the unit of X coordinate of axis2 is 'sample', and in offline mode the unit is second. This conversion is done by using MATLAB code `handles.datasample(:, 1).*(etime(clock,handles.starttime)/handles.datasample(end,1))` after you stopped the acquisition. As you may have seen, it just scales the array using the elapsed time (time between clicking the 'Start-Stop' button) and total number of samples, and it is not accurate, either. Axis2 is designed for monitoring the patch condition, but not for providing accurate time-current relationship. If accurate time-current relationship is desired, please use Capmeter1v3 or Capmeter6v3.

Giving pulses and applying notes



The pulse protocol used in IQplot is different from that used in Capmeter in several ways. First, pulse protocol in Capmeter is a one-way scanning protocol, but in IQplot, it is a loop-like protocol as shown above (i.e. the same voltage is applied twice in the entire protocol). The advantage of this kind of protocol is that one can examine if hysteresis exists, which usually happens when the charging time is not long enough (according to my mentor). Second, the pulses in IQplot are continuous. The voltage will not go back to

zero between pulses, and there is no interval between pulses, either. Third, unlike Capmeter, you can not give identical pulses in IQplot. Since IQplot is used for plotting I-V and Q-V, it doesn't make sense to give pulses with the same amplitude.

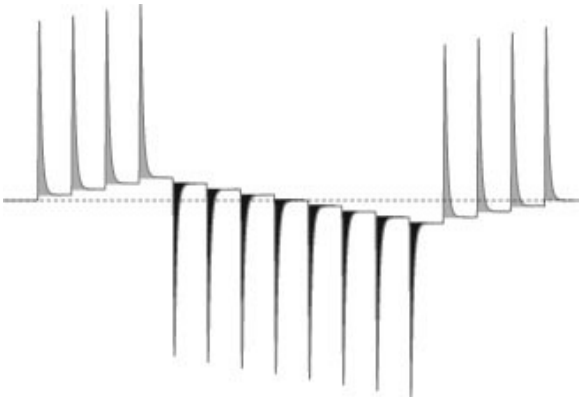
The values in the first two edit boxes in the 'Pulse protocol panel' define the voltage range you are going to scan. The sign of the values must be opposite (the program will make it different anyway), and the program will increase (or decrease) the voltage from zero toward the direction defined in the first edit box. You can define the length of each step, the size of the voltage step, and how many rounds you want to apply the protocol in the following edit boxes. You can always move the mouse cursor onto the edit boxes for few seconds to get the definition of each edit box. Clicking the 'Pulse' button triggers the pulse protocol. The program acquires 10 ms pre-trigger signals to estimate the constant DC current (e.g. caused by junction potential etc.) and then subtract it from the entire trace before further processing. When the protocol is completed, raw current (with pre-trigger signal), I-V, and I-Q plots are displayed on axis1, axis4, and axis5, respectively (axis3 is removed from IQplot2).

In addition to giving pulses by clicking the 'Pulse' button, you may also ask the program to apply the same protocol automatically in a defined time interval. To do so, you simply check the check box in the 'Pulse protocol panel' when the program is running, and IQplot will trigger the acquisition automatically. The pulse protocol is also triggered when you click the 'Set baseline' button in the 'Pulse protocol panel'. After the baseline is set, you may not change the pulse protocol, and the baseline records are subtracted from upcoming records if the 'Subtract' button is clicked. You may change the

pulse protocol again once the 'Set baseline' button is unclicked.

To apply notes, simply put the note in the first edit box in the 'Pulse note panel', and it will be associated with the next pulse automatically. For your convenience, there are three more edit boxes in the panel. You can put notes that are frequently used in these edit boxes. By clicking one of the 'Apply' buttons next to the edit box, the note will be placed in the first edit box and associated with the next pulse as mentioned previously.

Methods for charge integration



The current is not used for integration directly. For each pulse, the steady-state current (estimated using Eq. 3.2) is subtracted from the current, and the area shown in Fig. 4.6A is the integrated region. Since the steady-state current is subtracted, the integrated charges are independent of the seal condition. As shown above, the gray areas are added into the transferred charges, and the black ones are subtracted. The corresponding Q-V relationship is displayed on axis5.

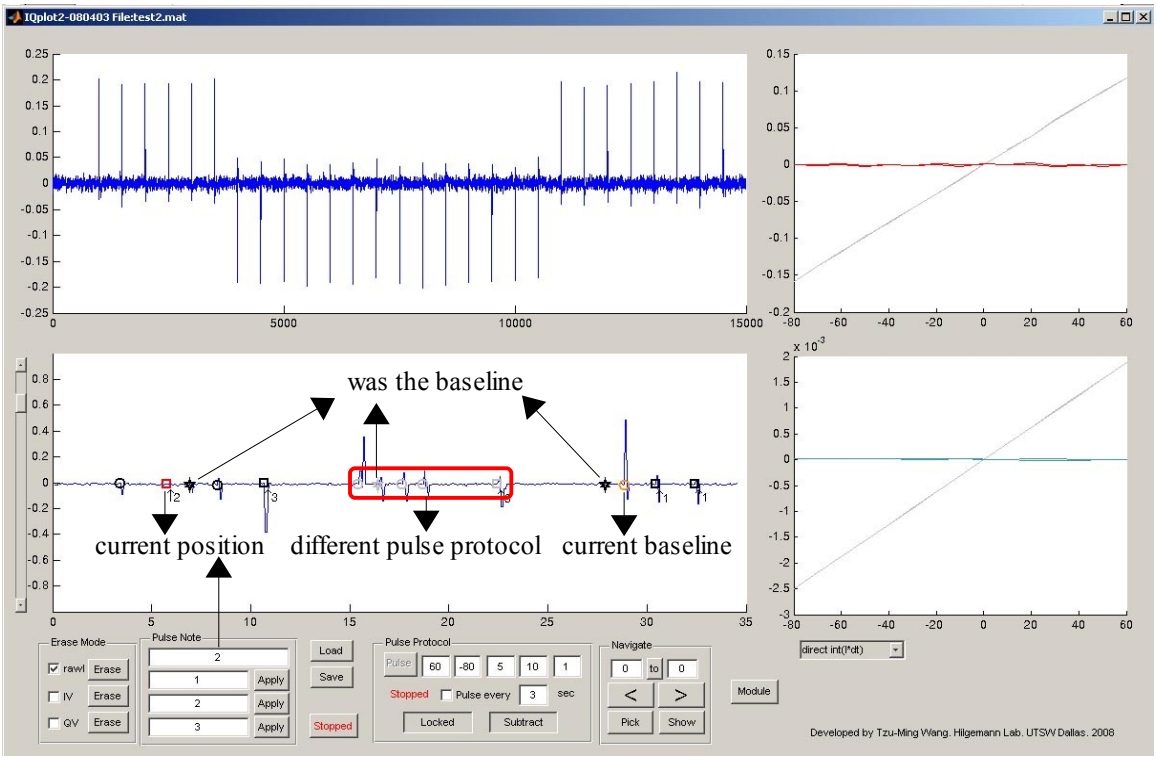
There are two methods for charge integration in IQplot2, and you may use either one of them in your experiment. The first method summates all $I_t\Delta t$ fragments and outputs the

final value. The second method further estimates the asymptote of the integrated charges (Fig. 4.6B) using Eq. 3.2, which is also used in Q-SQA. However, note that in Q-SQA the amount of transferred charges is corrected according to the time constant using Eq. 4.34. In IQplot, the value is not corrected because time constant from a single curve (every curve is considered different from each other) is not quite accurate, and will introduce excessive noise to Q if Eq. 4.34 is used. Also, these two methods are interchangeable during the experiment, so you might want to make a note to specify the method you used. The program does not keep track of it.

Display modes – online

When the recording is started, axis2 shows you the sampled current. After the first pulse protocol is given, axis1 shows you the corresponding current (the number of points displayed is reduced 10X to relieve CPU load in online mode), and axis4 and axis5 show I-V and Q-V plots, respectively. If the pulse protocol(s) is given more than once, the old traces will be shown in gray, and the new one is placed on top of them unless the 'Erase mode' is selected. To erase the old traces, you may either click the corresponding 'Erase' button in the 'Erase mode panel' every time when needed, or you may check the corresponding check boxes in the panel, and the old traces will be erased every time when the new one is plotted. You may adjust the scale of axis2 by dragging the slider bar next to it.

Display modes – offline



The online and offline display modes are very similar, except that you may adjust the scale of X axis of axis2 offline, and there are many symbols plotted on axis2 in offline mode. Each symbol represents a pulse train, and the current pulse train displayed is in red color. If there is a note associated with the current pulse, the note will also be displayed in the first edit box in the 'Pulse note panel'. Symbol for pulses with notes associated with them are in square shape. I would like to emphasize that the recording of variable *handles.starttime* may be delayed occasionally and makes the estimated time for axis2 very inaccurate. It should not happen theoretically, but as you may know, all kinds of unexpected conditions and/or errors may occur in MATLAB... Anyway, axis2 (and the corresponding variable *datasample* in the Workspace) should not be considered as a real experimental data. It is just a reference for you to monitor the seal condition etc. as

mentioned at the very beginning.

If you have clicked the 'Set baseline' button in the 'Pulse protocol panel' during the experiment, the given pulse train will be represented by a star-shaped mark in the offline mode. You may also set the baseline offline simply by clicking the 'Set baseline' button. Data for current pulse train will be used for subsequent subtraction, and the color of the mark becomes yellow. Note that you may set the baseline to data from any pulse train in the record, not just to one that has been used during the experiment. When the baseline is set offline, you can only access pulse trains which used the same pulse protocol as the baseline used. The color of symbols for pulses using different protocol(s) becomes gray in this case.

You may navigate the pulse trains by using the 'Navigate panel'. Click the '>' button to see the next record, and click the '<' button to check the previous one. You can use the 'Pick' button to jump to the desired record directly, too. You may also specify the time range in the edit boxes and then press 'Enter', or use the 'to' button to select a region to be shown on axis2. To show all the data on axis2 again, you simply put '0' (zero) in the edit box on the right side of the 'to' button and then click 'Enter'.

By clicking the 'Show' button in the 'Navigate panel', IQplot2 generates figures for all the axes, and the data are assigned to corresponding variables in the Workspace. Although you may specify the region to be displayed on axis2, the figure for axis2 still contains all the points (i.e. 1~end).

Variables in the Workspace

After the recording is stopped, IQplot assigns a number of variables to the MATLAB Workspace. They are introduced below.

DAQinfo, structure containing hardware settings

aiSR, analogue input speed, in Hz

aoSR, analogue output speed, in Hz

aoCh1convert, command sensitivity for AO0, in mV/V

starttime, approximate time when the 'Stopped' button is clicked, in [year month date hr min s]

Fig1Y, it contains Y data from axis1 when the 'Show' button is clicked

Fig2, it contains XY pairs of axis2 data when the 'Show' button is clicked

Fig3X, *Fig3Y*, voltage and current data from axis4 when the 'Show' button is clicked

Fig4X, *Fig4Y*, voltage and charge data from axis5 when the 'Show' button is clicked

Pulselog, structure containing pulse protocol and other information

eventcode, 0, no event; 1, setting baseline; 2, applying note

pulse, voltage values of steps in the pulse protocol (mV).

pulseinfo, [first peak (mV), second peak (mV), step length (AO points), step increment (mV), number of pulses, rounds]

note, affiliated note

trigger, AO trigger time, in [year month date hr min s]

dataindex, the last voltage step in the protocol is the *dataindex*th step in variable *data*

axis2index, pulse is given when there are *axis2index* data points in variable *datasample*

rawfileinfo, indexes for locating pulse records in raw files, in [file number, trigger number]

data, array containing processed data, in [*I*, *Q* (no summation, yet), τ]

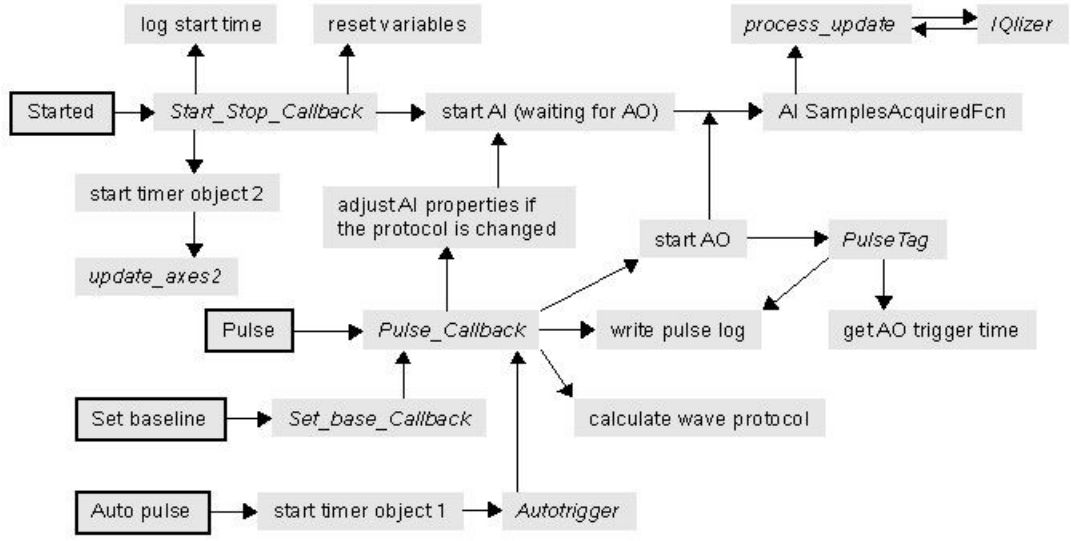
datasample, sampled current. It does not represent accurate time-signal relationship

version, structure containing version information

Shell, version of the GUI

Engine, version of the *IQlizer*

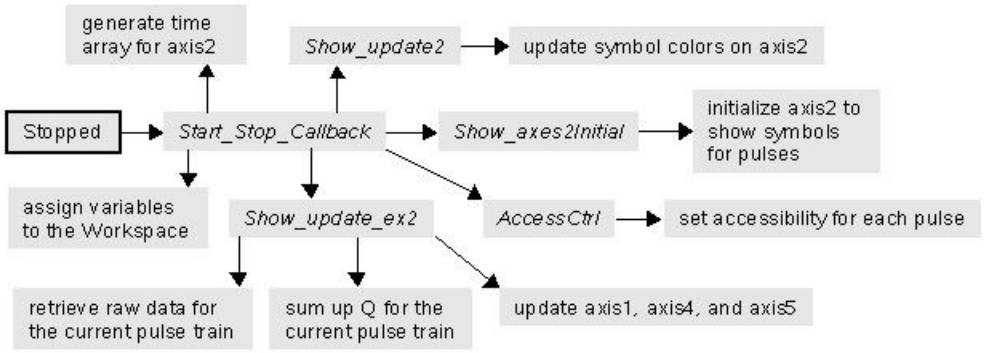
The information flow



This section and the following one introduce how IQplot2 works. The above diagram shows you the information flow when the 'Stopped' button is clicked. Function names are in italic, and names of buttons and check box are in black-square text boxes. When you start the recording, function *Start_Stop_Callback* is evoked. It resets some variables and adjusts properties of GUI components. Variable *handles.starttime* is logged here, and AI and timer object 2 are then started. Timer object 2 evokes its *TimerFcn* every 50 ms to refresh axis2. Once the AI object is started, it is waiting for the trigger signal (i.e. the first pulse) from AO. There are three ways to trigger the AO object when the program is running. The first one is to press the 'Pulse' button, the second one is to click the 'Set baseline' button, and the last one is to check the 'Auto pulse' check box. When the 'Auto pulse' check box is checked, timer object 1 is started and its *TimerFcn* calls function *Pulse_Callback* periodically to trigger the AO object. The *Pulse_Callback* does several things. When it's evoked, it first compares the current pulse protocol with the previous

one. If the two protocols are different, it generates new AO data for the current pulse protocol, and adjusts AI properties *SamplesAcquiredFcnCount*, *SamplesPerTrigger*, *LogFileName*, *TriggerCondition*, and *TriggerConditionValue*. If the protocols are the same, the original AO data and AI settings are used. Besides calculating the pulse protocol and adjusting AI properties, the function also prepares information for *handles.Pulselog*. Once everything is ready, the function queues AO data into memory and then trigger the AO object. The *TriggerFcn* of the AO object (*PulseTag*) records the initial trigger time of each pulse, although this information is not used anywhere else in IQplot2.

The AI object is triggered by the first pulse in the protocol. Once the desired number of samples (calculated and set in *Pulse_Callback*) are acquired, function *process_update* is evoked. This function calls *IQlizer* to process the data and shows raw data and processed data in the corresponding axes. If the *Pulse_Callback* is evoked by clicking the 'Set baseline' button, the *process_update* function also sets the *eventcode* in *handles.Pulselog* to 1 as a reference. Note that although the time constant is not used for any purpose in this program, the *IQlizer* still calculates and returns time constant for each pulse step. This feature is reserved for future development of the program.



There are still many things to deal with after the recording is stopped. Besides assigning variables to the Workspace and generating time array for axis2, the *Start_Stop_Callback* calls function *Show_axes2Initial* to initialize the offline display mode for axis2. The function *AccessCtrl* compares all the pulse protocols in the *Pulselog* to the current protocol used for baseline subtraction. If the protocol is different from the baseline protocol, the symbol for the pulse train is set to gray, and you will not be able to access it unless the 'Set baseline' button is unclicked. Once the initialization of axis2 is completed, function *Show_update2* is called. This function simply sets the symbol color for current pulse train to red, and sets symbol color for current baseline to yellow if the 'Set baseline' button is clicked. Function *Show_update_ex2* updates all the axes except axis2. It retrieves raw data using MATLAB function *daqread*, subtracts the data if desired, sum up charges from each individual pulse response, and then update the plots.

Main variables in the program

As usual, there are tons of variables in the program, and most of them are categorized into groups and have clear descriptions in the codes. A few important variables are introduced below, and many of them are also used in Capmeters.

handles.bufdir, it contains temporary directory used to save raw files. The default names for the raw files are IQraw-numbers.daq. The raw files will be renamed to FILENAME-numbers.daq and moved to the directory indicated by *handles.current_folder*.

handles.current_folder, it is the last folder you visited when you use the 'Save' or 'Load' button.

handles.nidaqid, device ID for the data acquisition board. Please refer to section 4.3 for more details.

handles.aoCh1convert, command sensitivity for AO0, in mV/V.

handles.aiSR, acquisition speed for AI, in Hz.

handles.aoSR, acquisition speed for AO, in Hz.

handles.aidata, n-by-3 matrix containing all the processed data.

handles.aodata, column array containing output signals for AO0.

handles.datasample, current sampled at 20 Hz using MATLAB function *peekdata*.

handles.starttime, the time when the 'Stopped' button is clicked.

handles.TRIGGERDELAY, the default trigger delay value. Negative value is used so that the pre-trigger signals are acquired.

handles.triggerdelay, it is the trigger delay value used for the current record.

4.6 Dynamically linked subroutines

CapEngine4.mexw32

CapEngine4 is the core processing unit of Capmeter6v3. It performs all the required calculations to generate digitized data. These functions include 1. assigning relative time to every data point, 2. performing background digital filtering for Ch3 (AI1) and Ch4 (AI2), 3. performing PSD, 4. performing I-SQA, and 5. performing Q-SQA. The syntax for using PSD in MATLAB is,

```
[time,Ch3,Ch4,Ch1(Y),Ch2(X)] =
CapEngine4(1,time,AI1,AI2,fck3,fck4,filterpt,aiSamplesPerTrigger,PSDref
,PSD90)
```

The first input argument is 1, which tells *CapEngine4* to use PSD (2 for I-SQA and 3 for Q-SQA). Input arguments *fck3* and *fck4* determine whether Ch3 and Ch4 are digitally filtered or not, and argument *filterpt* is the number of points used for averaging. Argument *aiSamplesPerTrigger* tells *CapEngine4* the length of each trigger, and arguments *PSDref* and *PSD90* are in-phase and orthogonal reference waves, respectively.

```
[time,Ch3,Ch4,Ch1(Y),Ch2(X),asymptote,peak,tau] =
CapEngine4(2or3,time,AI1,AI2,fck3,fck4,filterpt,aiSamplesPerTrigger,PSD
ref,PSD90,AI0,(optional)taufactor,(optional)endadj)
```

To use SQA, the first input argument has to be 2 or 3, and there are three more input arguments. Argument *AI0* is the trigger signal used for locating each individual curve. Arguments *taufactor* and *endadj* are optional. The *taufactor* is used for determining the region for curve fitting (Fig. 4.5B, solid section). Empirically, *taufactor* of 3 is used for I-SQA, and 1 is used for Q-SQA. If you do not specify the *taufactor*, the entire curve (from

the peak to the end) will be used for curve fitting, which is not recommended. The last input argument is *endadj*. It is used for further adjusting the fitted region in earlier versions of *CapEngine*, and the default value in *CapEngine4* is zero. It was -5 in Capmeter6v3. That means if *taufactor* determines that the 30th data point is the end of the fitted trace, for example, *endadj* tells *CapEngine4* to move the boundary by -5 points, and the 25th data point is the real end of the fitted trace. This argument is no longer critical in the present algorithm, and may be removed from the MATLAB code in the future.

There are three more output arguments for SQA. They are steady-state current (asymptote), peak current, and time constant. Note that even when SQA is selected, *CapEngine4* still runs PSD for the input signals. The returned PSD results are assigned to variable *handles.PSDofSQA* in Capmeter6v3.

Dfilter.mexw32

Dfilter is the background averaging filter used in Capmeter1v3 and its function has been integrated into *CapEngine4*, which is used for Capmeter6v3. It can process multiple channels, and the syntax is,

```
output = Dfilter(fcks,matrix,aiSamplesPerTrigger,filterpt)
```

Argument *fcks* is a 1-by-n array that tells *Dfilter* if the channels are going to be filtered or not. Argument *matrix* is a m-by-n matrix, where 'm' is the total number of points for each channel, and 'n' is the number of channels. For example, if *fcks* is [1,0,0,1], and the *matrix* is [Ch1,Ch2,Ch3,Ch4], all the channels are digitized according to *aiSamplesPerTrigger*, but only Ch1 and Ch4 are filtered (using *filterpt* points). The output

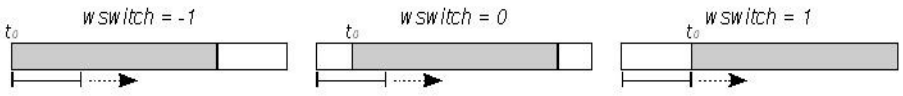
is a p-by-n matrix where 'p' is equal to (m+1)/(*aiSamplesPerTrigger*+1) (+1 is the NaN).

Dfilter2.mexw32

Dfilter2 is a running mean/median filter used in Capmeters. The syntax is,

```
output = Dfilter2(fswitch,data,fwindow,(optional)wswitch)
```

Input argument *fswitch* is a filter switch that tells the function what to do. Values of 0, 1, and 2 indicate bypassing the filter, running average filter, and running median filter, respectively. Argument *data* must be a m-by-1 array (processes 1 channel at a time), and the output argument is also a m-by-1 array. Input argument *fwindow* is the window size used for filtering, and *wswitch* defines the way *Dfilter2* handles the data-time relationship.



As shown above, in order to keep the dimension of the output data the same with the input (gray), *Dfilter2* adds *fwindow*-1 points (white) to the original data set. If *wswitch* is -1, *Dfilter2* repeats of the last data point *fwindow*-1 times at end of the input data, and t_0 is on the left-hand side of the filter window (solid section). For *wswitch* of 1, the first data points are repeated *fwindow*-1 times at the beginning of the data set, and the t_0 point is on the right-hand side of the window. The default value for *wswitch* is 0, that means the first and the last data points are repeated (*fwindow*-1)/2 times at the beginning and the end of the data, respectively, and the t_0 point is in the middle of the window. If *fwindow* is even, let's say 8 for example, the first point is repeated 4 times, and the last data point is

repeated 3 times.

To get the median, *Dfilter2* uses build-in C function *qsort* to sort the data, and then get the median. If the window size is even (e.g. 20), *Dfilter2* takes the average of the middle 2 sorted points (i.e. 10^{th} and 11^{th}) as the median. The *qsort* function uses Quicksort sorting algorithm, and you may find the introduction about it at http://en.wikipedia.org/wiki/Qsort.

Dfilter2 does not do summation/sorting for every loop, instead, it only does it once in the first loop. For subsequent loops, *Dfilter2* removes one old data point from the sum/series, and adds the new one to the sum (for running mean) or to the appropriate position of the series (for running median). In this way, the processing speed of *Dfilter2* is greatly enhanced. The calculation of running mean becomes superfast, and the running median filter becomes at least 75 times faster in my computer.

For data containing NaN, *Dfilter2* removes NaN in the window and then calculate the mean/median. If there is no real number in the filter window, NaN is assigned to that time point. Since *Dfilter2* removes NaN before calculation, the total number of NaN will be reduced after filtering. For instance, if the longest consecutive NaN in the input data set is less than the filter window, there will be no NaN in the output array. The original data gaps (composed of NaN) are filled by data average/median surround the gaps.

DispCtrl.mexw32

Because showing data takes a lot of CPU power, Capmeters use function *DispCtrl* to reduce the displayed data in the online (only called in AI *TimerFcn*: *update_plot*) but not

offline mode. It simply samples equally-spaced points from the input array, and the space is determined by the total length of the data and the number of points to be displayed. If the number of total points is less than the setting, all points are displayed. An example is shown below.

```
[XData12,YData1,YData2,XData3,YData3] =
DispCtrl(15000,XData12,YData1,YData2,7500,XData3,YData3);
```

The above example restricts at most 15000 points to be displayed on top and middle panels, and at most 7500 points to be displayed on the bottom panel.

IQlizer.mexw32

Like *CapEngine4* for Capmeter6v3, *IQlizer* is the core processing unit of IQplot2. It is modified from the subfunction *SqQ* in *CapEngine4*. The main difference is that *IQlizer* does not correct the integrated charges with the time constant as mentioned previously. The syntax for *IQlizer* is,

```
[asympote,Q,tau] = IQlizer(AI1,time,pulse_L,pulse_NR,(optional)mode,
(optional)taufactor,(optional)endadj)
```

Argument *pulse_L* is the number of AI points for each step, and *pulse_NR* is the number of total steps. Input argument *mode* tells *IQlizer* whether to output the direct summation of the charges (default, *mode* = 1), or to output estimated total-transferred charges at steady-state (*mode* = 0). Arguments *taufactor* and *endadj* are used the same way as in *CapEngine4*.

Output arguments include estimated steady-state current (*asmyptote*), integrated charges (*Q*), and estimated time constant (*tau*). Note that *Q* is a column array containing

integrated charges for each individual curve (*asymptote*-subtracted). To plot Q-V, IQplot2 add them up right before axis5 is updated.

PhaseMatcher2.mexw32

PhaseMatcher2 is used in Capmeter6v3 when PSD analysis of SQA data is desired. Its two functions are, 1. looking for the optimal phase angle using cross correlation and 2. calculating correlation coefficient between two input traces. For detail, please refer to section 4.2 'Square-wave perturbation – PSD analysis of SQA data'. To scan the phase spectrum, the syntax is,

```
[P,C_sft,G_sft,R_c,R_g]                                                    =
PhaseMatcher2(switch,Csqa,Gsqa,Cpsd,Gpsd,step(degree))
```

Input argument *switch* determines the channel(s) used for cross correlation. If *switch* = 1, *PhaseMatcher2* returns an angle so that PSD and SQA from Ch2 (conductance) have the highest correlation. If *switch* = 0, PSD and SQA from Ch1 (capacitance) are used. For *switch* < 0, both Ch1 and Ch2 are used, and the sum of cross coefficients between PSD and SQA from Ch1 and Ch2 is the highest at the given angle. Input arguments *Csqa*, C*psd*, G*sqa*, *Gpsd* denote capacitance from SQA, from PSD, conductance from SQA, and from PSD, respectively. The last argument *step* defines the scanning step size (in degree) in *PhaseMatcher2*. Scanning step size of 0.1 degree is used in Capmeter6v3. The output argument *P* is the obtained phase angle, and *C_sft* and *G_sft* are phase-shifted Ch1 and Ch2 data at the given angle *P*, respectively. Output arguments *R_c* and *R_g* are the corresponding correlation coefficients at angle *P*.

To get the correlation coefficient between two traces, simply use the following syntax,

```
R = PhaseMatcher2(trace1,trace2)
```

The correlation coefficient (*R*) is returned.

SqWaveCalc.mexw32

The only function of *SqWaveCalc* is to generate AO data for square wave perturbation in Capmeter6v3. The reason to make a C rather than a MATLAB function is because *SqWaveCalc* might be evoked frequently when the SQA 'auto frequency' option is selected. Generating a wave form takes a lot of loops in the program, and MATLAB does not handle loops efficiently. If the wave form is not generated in time, the acquisition is jammed. That is why the function is in C.

```
output = SqWaveCalc(totalAOpt,SamplesPerWave,amplitude)
```

Input argument *totalAOpt* is the total number of AO points per trigger. It is determined by AO speed (*handles.aoSR*) and the recording frequency (*handles.rSR*). Argument *SamplesPerWave* tells the function the number of AO points for a single wave. It is determined by AO speed and oscillation frequency. Note that *totalAOpt* has to be the multiple of *SamplesPerWave*, and the trigger signal is added in MATLAB function *Wavecalc* but not in C function *SqWaveCalc*.

4.7 Capmodule4

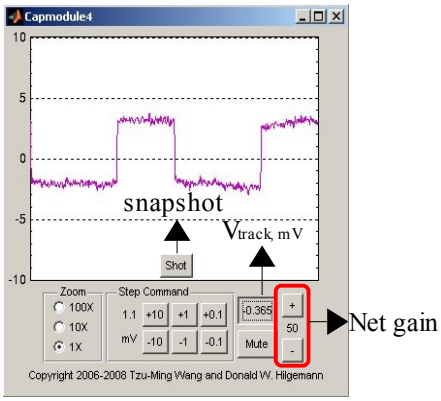
What is it?

Capmodule4 is a tool for you to make the seal (patch clamping). It generates step command, tracks the current, displays the current, and makes sound (noise) according to the current. Axopatch-200 (Molecular Devices) + Capmodule4 ~= Axopatch-1D.

Things you need to know before using

1. Please cite the paper: Wang and Hilgemann, 2008.
2. Please adjust the net gain (αβ) from the GUI
3. To save the snapshot data, please use the 'Save' button in the Workspace. The 'Save' button in Capmeter does not save snapshot data.

Running the program



The most common way to launch Capmodule4 is clicking the 'Module' button or pressing the '*' key in Capmeters. In this way, Capmodule4 gets the channel carrying current, device ID, and command sensitivity directly from Capmeters. Make sure to adjust the net gain using the '+' and '-' button. If you get the current from the scaled-

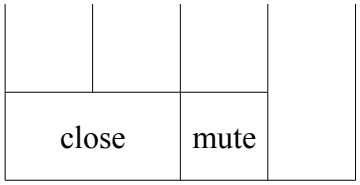
output of the patch clamp, the net gain is αβ. If the current is not scaled, the net gain is the gain of the headstage (β). The value of net gain influences the stability of the tracking function and the generation of the noise. Once the program is started, noise is generated immediately. If you do not like the noise, you may click the 'Mute' button to shot it off, or change the 'Value' property of 'Mute' button to 1 using MATLAB GUI builder.

Click the 'Track' button to keep the net current zero, and the tracking potential (in mV) is shown in the button. You may adjust the step command potential using corresponding buttons in the 'Step command panel', and the current step size is shown in the panel. To adjust the zoom, simply click the desired radio button to change the Y axis setting.

You may take snapshots of the oscilloscope. To do so, click the 'Shot' button as many times as you want, and the stored data will be assigned to the Workspace variable *shotdata* once Capmodule4 is closed. Variable *shotdata* is a m-by-2 matrix, and the first and the second columns represent command potential and current, respectively. If multiple shots are taken, NaN is inserted between the shots. To store the *shotdata*, you have to use the 'Save' button located at the top of the Workspace. Variable *shotdata* will be cleared if you use the 'Save' button in Capmeter.

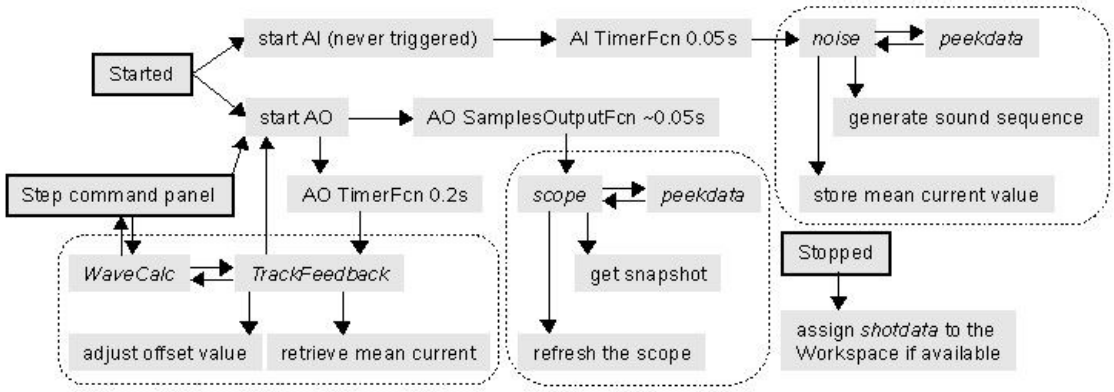
Hot keys

<table>
<tr><td>Num Lock</td><td></td><td></td><td></td></tr>
<tr><td>+10</td><td>+1</td><td>+0.1</td><td rowspan="2">track</td></tr>
<tr><td>-10</td><td>-1</td><td>-0.1</td></tr>
</table>



The above diagram shows you the keypad of the keyboard with designated functions. You may use the keypad to control the step command potential and other functions of the program. Make sure 'Number lock' of the keyboard is 'on' if you want to use the hot keys.

The information flow



The above diagram shows you the information flow in Capmodule4. Events happening periodically are in dashed boxes, and function names are in italic. Once the program is launched, AI and AO objects start automatically. Actually, AI is never triggered, however, its *TimerFcn* is still evoked periodically to generate the noise. Function *scope* locates the positions of the pulses, refreshes the oscilloscope and gets the snapshot when the 'Shot' button is clicked. When the 'Track' button is clicked, the *TrackFeedback* function adjusts the holding potential to make average current close to zero. *TrackFeedback* adjusts the holding potential in a stepwise manner (every 0.2 s), and the step size is determined by

the average current. The smaller the absolute value of the average current, the smaller the voltage step. If you feel the tracking is too slow (i.e. taking a long time), or too fast (i.e. jumping around), you may adjust the step size in *TrackFeedback*. Whenever the absolute value of the average current is smaller than a certain threshold (defined in the function), a negative feedback is applied to reduce the holding potential by 0.005 mV in every cycle. The reason to implement a negative feedback here is that when seal is formed suddenly, the average current is dropping close to zero, and in this circumstance, it does not make sense to keep the same magnitude of the holding potential. This design is more reasonable rather than critical.

The sound sequence

The sound sequence is composed of two components. One is the carrier wave, the other one is the DC-subtracted, weighted current. The carrier wave is a sine wave, and its frequency is determined by the average current. The pitch is higher when the current is more positive, and lower when the average current is more negative. The current-frequency relationship is a sigmoidal curve, and it is most sensitive when the average current is close to zero (i.e. it is most sensitive during seal formation). The amplitude of the carrier wave is enhanced if its frequency is lower than 1.5 kHz.

The input current wave form is also weighted around its average value using an exponential function. The purpose is to magnify little current steps, and help the users to “hear” the seal during seal formation. From the “quality” of the sound, users can judge the quality of the seal, and from the changing pitch of the sound, users will know if the seal is getting better or worse. When a seal is perfectly formed, you will hear clean, and

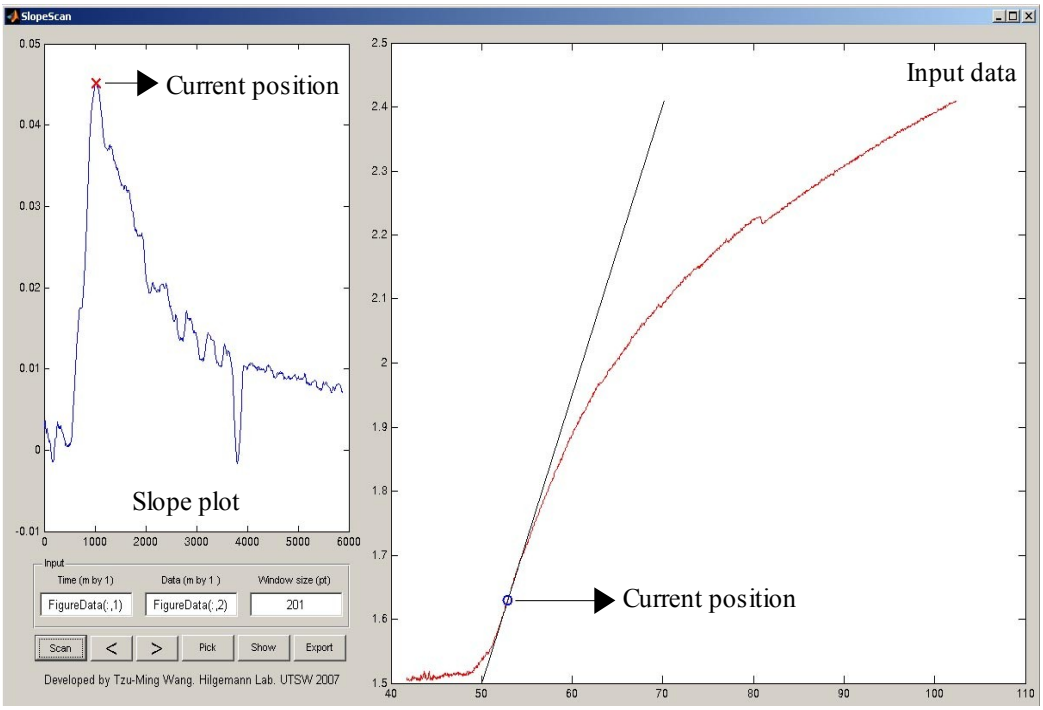
peaceful sound (else, annoying “noise”). If the sound response does not satisfy you, you may edit it in function *noise*. Note that the frequency the *noise* function is evoked also affects the quality of the sound. The frequency is determined by *handles.timerperiod*, and it also affects the refreshing frequency of function *scope*.

4.8 SlopeScan

What is it?

SlopeScan is used for finding the maximal slope of a trace. That's it.

How to use it?



You put the names of time and data arrays, and the window size used for slope calculation in the corresponding edit boxes, and then click the 'Scan' button. The slope along the input trace is shown on the left panel, and the original trace and tangent with maximal slope is shown on the right panel. And values of time and slope of the current position are shown in the MATLAB 'Command Window'. Note that SlopeScan retrieves input variables from the MATLAB Workspace, so both of the variables have to be present in the Workspace, and both of them have to be m-by-1 column arrays, too. The averaged time of the window is assigned to the output time array. That is, if the window size is odd,

the output time values are identical with the inputs.

Press '<' or '>' buttons (or left/down, right/up on the keyboard) to move along the trace, or use the 'Pick' button and then click on either of the panels to select the current position. You may generate figures by clicking the 'Show' button, and when the 'Export' button is clicked, variable *SlopeExported* is assigned to the Workscape. The format is [time,slope]. You may show/hide the tangent by pressing the 'h' key on the keyboard.

The scanning process is done by function *Rslope.mexw32*. *Rslope* uses the specified window size and linear regression to get the slope. It moves along the trace and assigns the corresponding slope to the output array. Unlike *Dfilter2* in Capmeters, *Rslope* does not add extra data points to the original input, so the output dimension will become smaller. For example, If input has 1000 data points and the window size is 11, the number of output points is 1000-11+1=990. The syntax for using *Rslope* is,

```
[newtime slope]=Rslope(time,data,window)
```

The input argument *time* has to be a m-by-1 column array. You can only use m-by-1 array for input argument *data* when you call *Rslope* from the SlopeScan, but you may use m-by-n column array when *Rslope* is called directly from the Command Window. In this case, *Rslope* will calculate running slope for each column, and the corresponding slope is assigned to the output column array *slope*, with dimension of p-by-n (p<m).

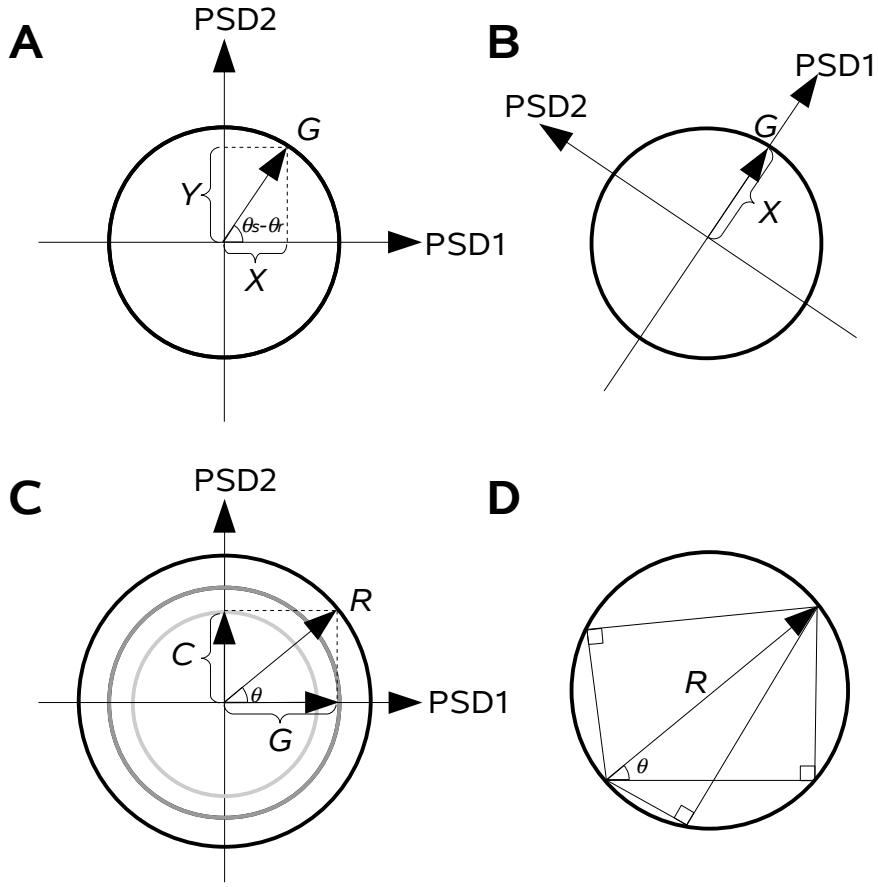


Figure 4.1. Geometrical view of phase-sensitive detection. A. If the phase angle is not properly adjusted, (i.e. θ_s-$\theta_r \neq 0$), according to Eq. 4.4, the readout of PSD1 is the projection of G on the x-axis (X), and G also contaminates the readout of PSD2 (Y), which is $\pi/2$ away from PSD1 (Eq. 4.5). To adjust the reference phase angle θ_r, it is just like rotating the coordinates by an angle of θ_s-θ_r, as shown in **B**. After the adjustment, X reflects the actual amplitude of G and nothing contaminates PSD2. **C.** When θ_r is properly adjusted, PSD1 and PSD2 give the actual G and C, respectively. The mixture of the two components is a new vector, with an amplitude of R and a phase angle of θ. It is important to note, however, by knowing R and θ does not mean that you can extract the actual G and C. There are infinite ways to get a vector with R and θ, and three of them are shown in **D**.

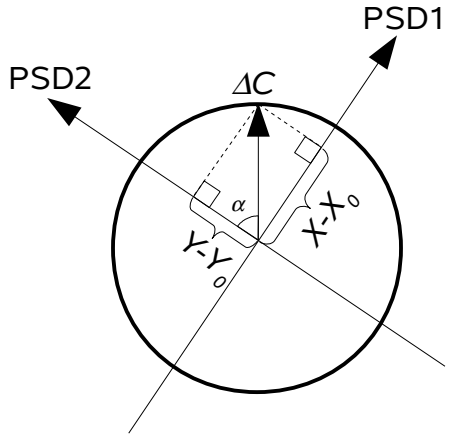


Figure 4.2. Online calculation of the phase angle. To test whether the current θ_r is valid or not, one can change the amount of compensated capacitance from the patch clamp, and see if the change (ΔC) is solely reflected on the readout of PSD2. If θ_r is not correct, changes of capacitance compensation result in signal changes both on PSD1 (X-X_0) and PSD2 (Y-Y_0), as shown above. The correct phase angle can be calculated by rotating the PSD1-PSD2 coordinate by an angle of α, given that α is $tan^{-1}((X-X_0)/(Y-Y_0))$ and X_0 (Y_0) and X (Y) represent values before and after changing capacitance compensation, respectively.

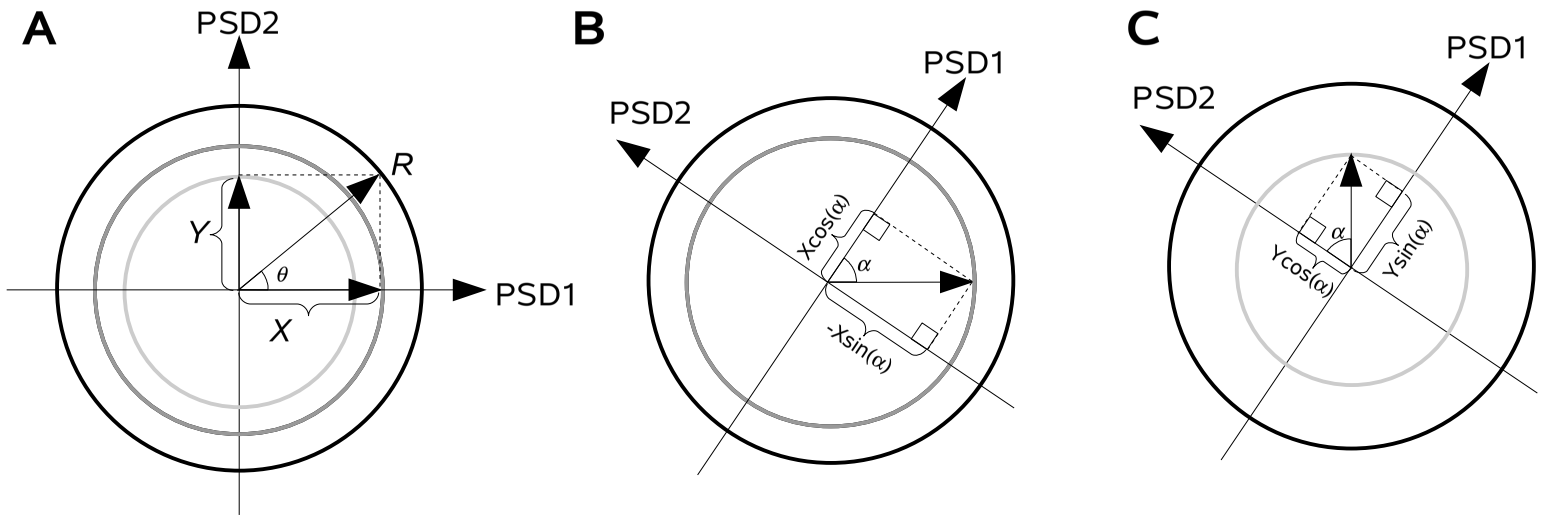


Figure 4.3. Offline adjustment of the phase angle. To adjust the phase angle offline, one can rotate the PSD1-PSD2 coordinate with a desired phase shift (α) to reconstruct a new set of *X* and *Y*. **A**. Considering that what the lock-in amplifier sees are two orthogonal vectors (*X* and *Y*) that compose a vector with *R* and θ. By rotating the PSD1-PSD2 coordinate, the new PSD1 readout is the sum of the projections of the original *X* (**B**) and *Y* (**C**) on the PSD1 axis, which is $X\cos(\alpha)$ and $Y\sin(\alpha)$, respectively. Again, the new PSD2 readout is the sum of the projections of the original *X* and *Y* on the PSD2 axis, which is $-X\sin(\alpha)$ and $Y\cos(\alpha)$, respectively.

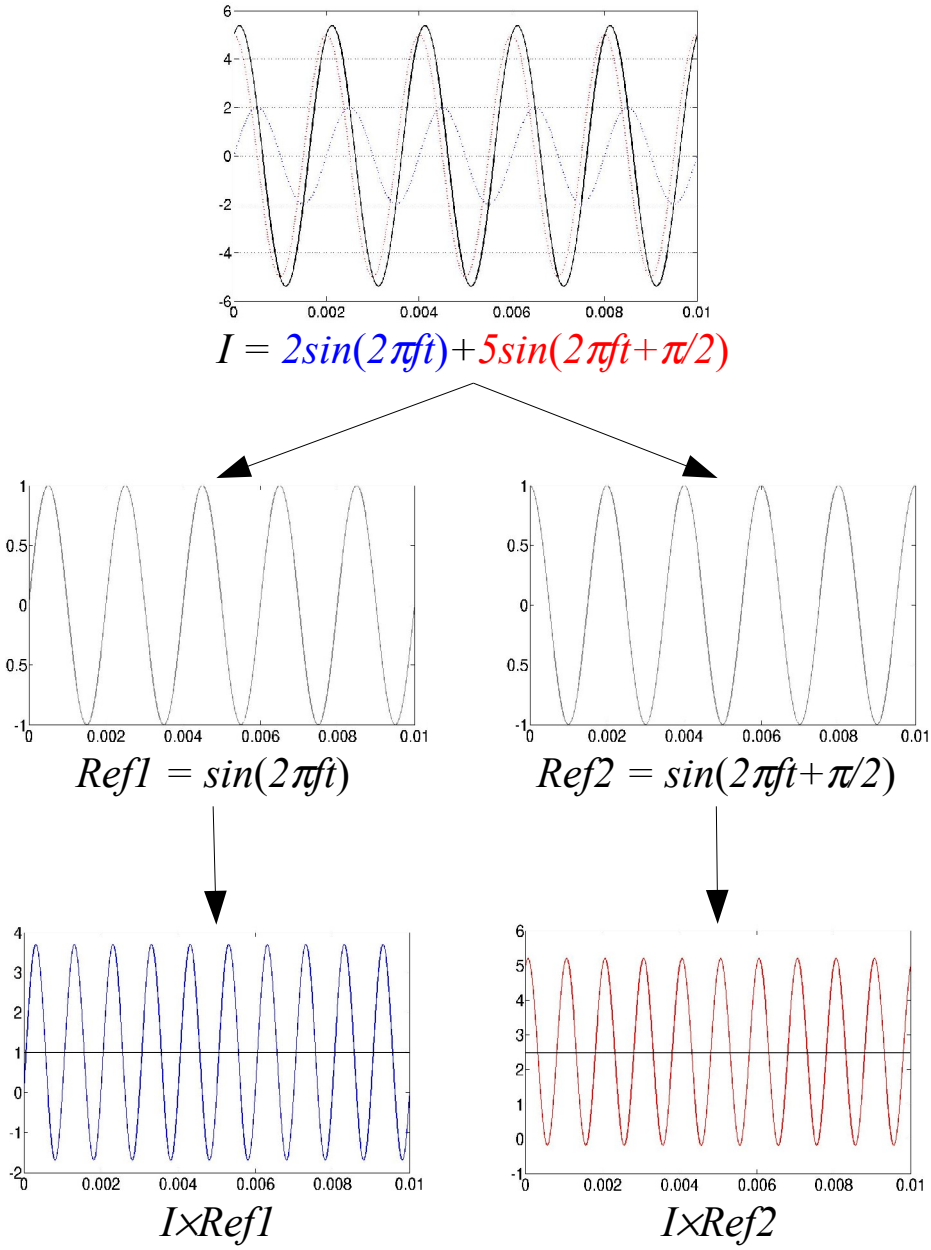


Figure 4.4. Demonstration of phase-sensitive detection. Current (*I*, black) composed of conductive (blue) and capacitive (red) components are shown on the top. In-phase (*Ref1*) and orthogonal (*Ref2*) reference waves are shown in the middle. The products are shown on the bottom. The value of the DC component (black line) is exactly half of the original signal amplitude.

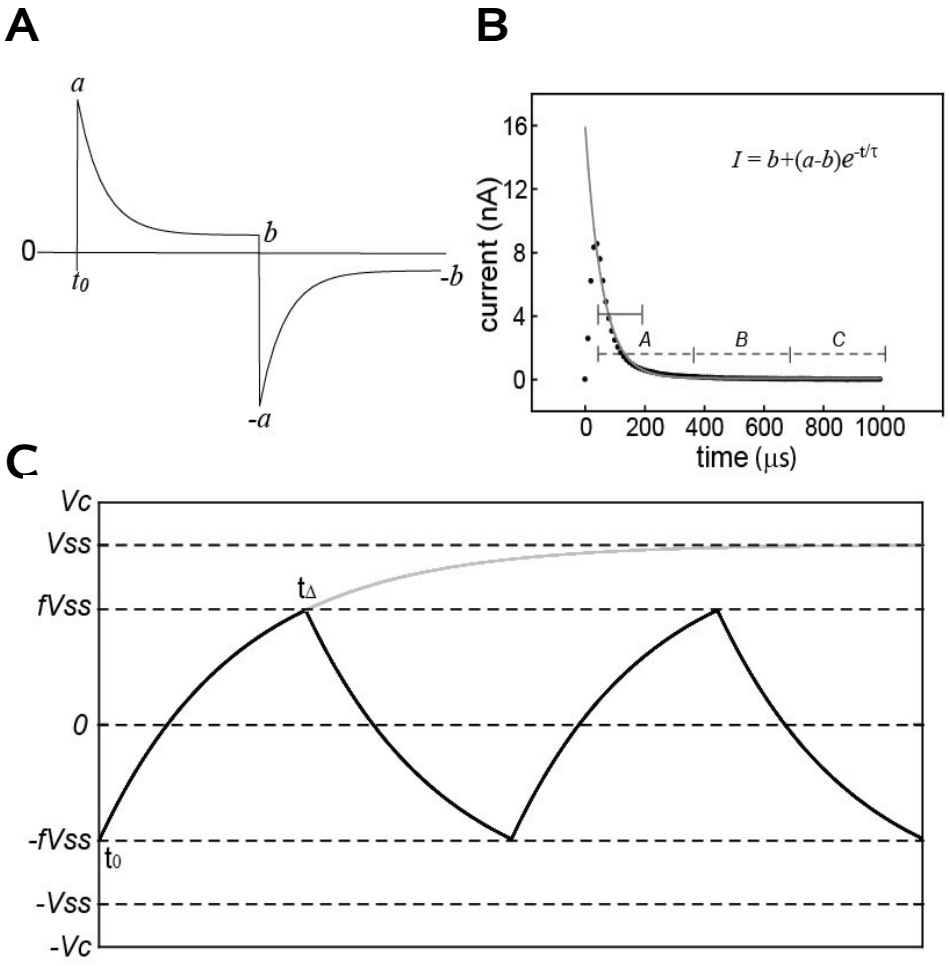


Figure 4.5. Square-wave perturbation – based on fitting the current transient. Model current for whole-cell recording is shown in **A**. Peak current, *a*, and the projected steady-state current, *b*, were determined as described in the text. **B.** Half of the current from a real recording (dots) with the fitted exponential function is shown. The asymptote of current was determined using averages of dashed sections according to Eq. 3.2. Data range from the peak to a point located at ~3τ was used to determine the exponential constants (solid section). **C.** Membrane potential during fast voltage oscillation (±*Vc*) is shown. Because the oscillation is too fast, membrane potential is not able to reach its theoretical steady-state value (±*Vss*), and oscillating between ±*fVss*, where *f* is the fraction of *Vss* across the membrane at the end of the voltage step of duration, t_{Δ}. Please refer to the text for some more details and equations.

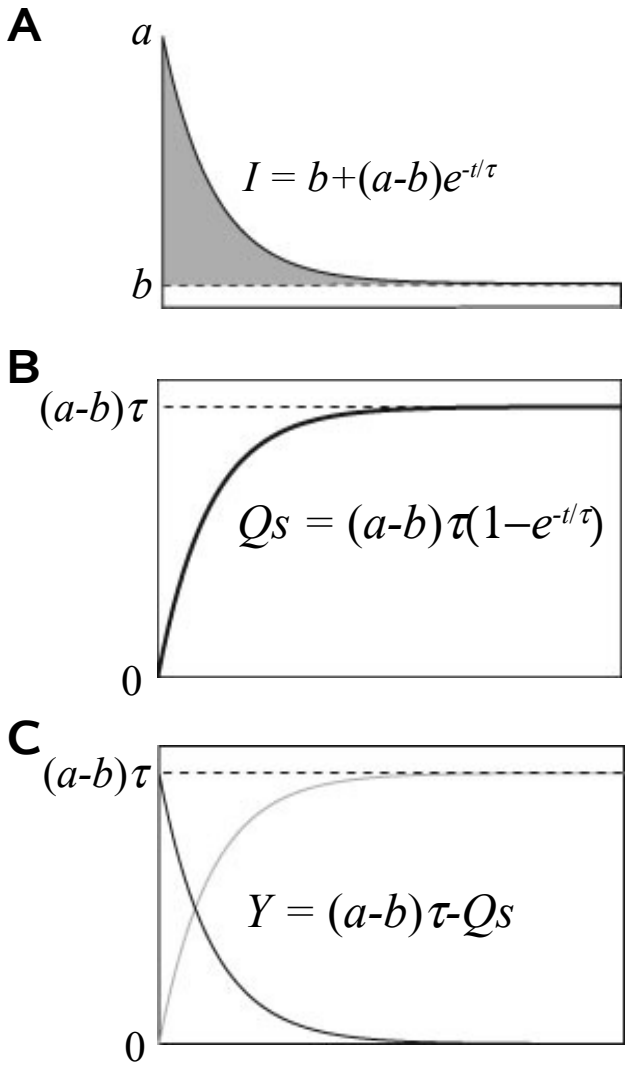


Figure 4.6. Square-wave perturbation – based on fitting the transferred charges. Half of the model current for whole-cell recording is shown in **A**, with peak current, *a*, and the projected steady-state current, *b*, which is obtained using Eq. 3.2. The gray area is integrated and the resulting charge (*Qs*)-time relationship is shown in **B**. The steady-state *Qs* is also estimated using Eq. 3.2. **C.** To get the peak current (*a*) and time constant (τ), the trace is inverted first, and linear regression of time against the natural logarithm of *Y* gives τ and (*a*-*b*)τ. Since *b* and τ are known, *a* can be obtained accordingly.

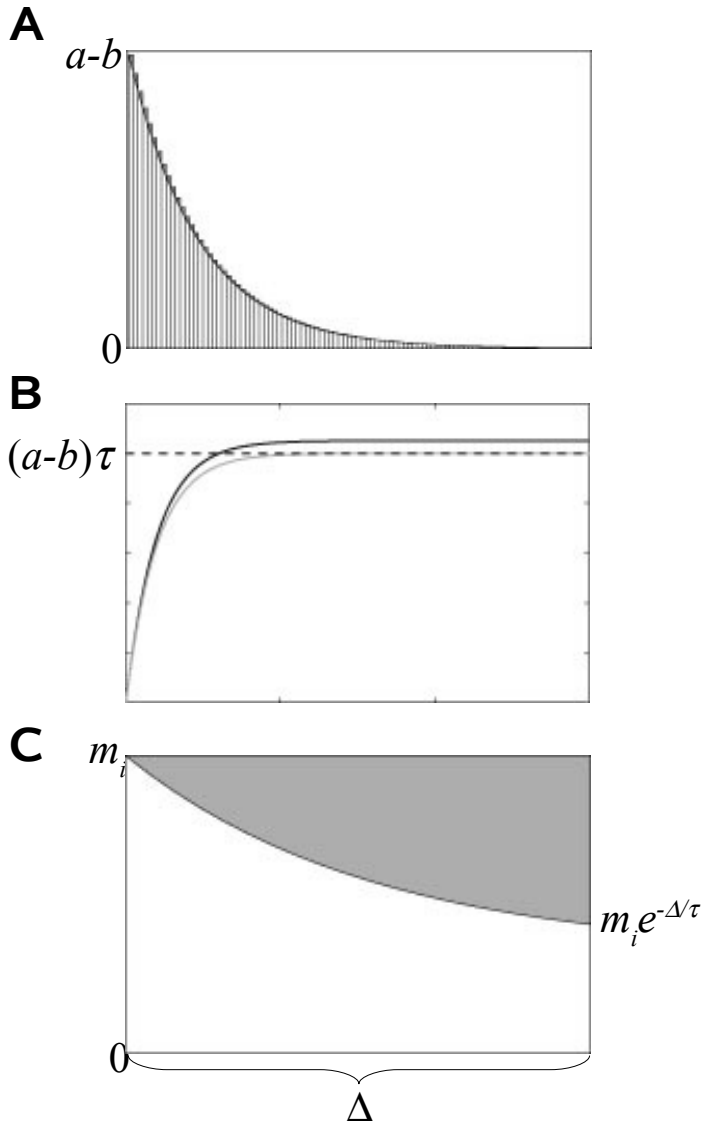


Figure 4.7. Correction of the integrated charges. Direct summation of the product of the current and time interval will over estimate the transferred charges because of the gray areas as shown in **A**. The phenomenon is simulated in MATLAB and shown in **B**. The back trace shows the summation, and the gray trace represents the actual transferred charges. **C.** Current acquired in an acquisition interval of Δ is shown. The ratio of the square and the white area is used to correct the transferred charges (Eq. 4.34).

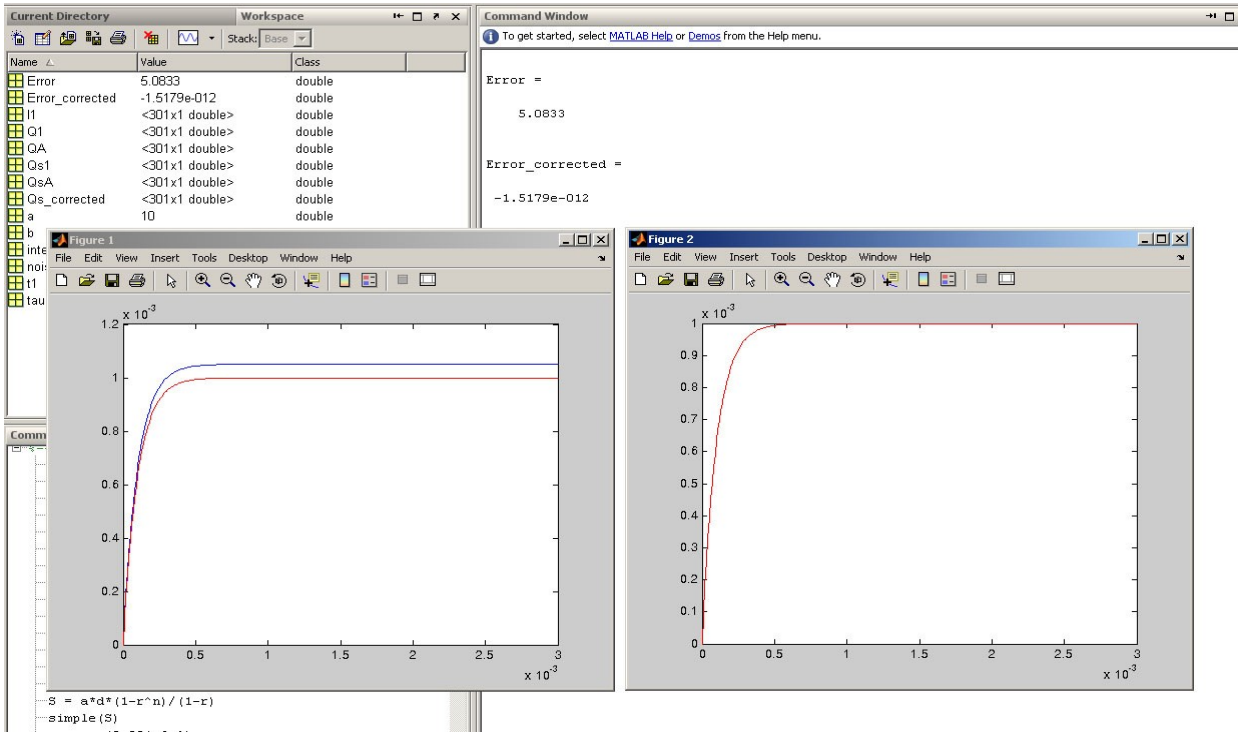


Figure 4.8. Demonstration of the charge correction. Screen shot of the MATLAB. In the left panel, the theoretical trace is in red and the direct charge summation is in blue. After correction using Eq. 4.34, two traces overlap each other as shown in the right panel. The errors are ~5% and ~-1.5*10^{-12}% before and after correction, respectively.

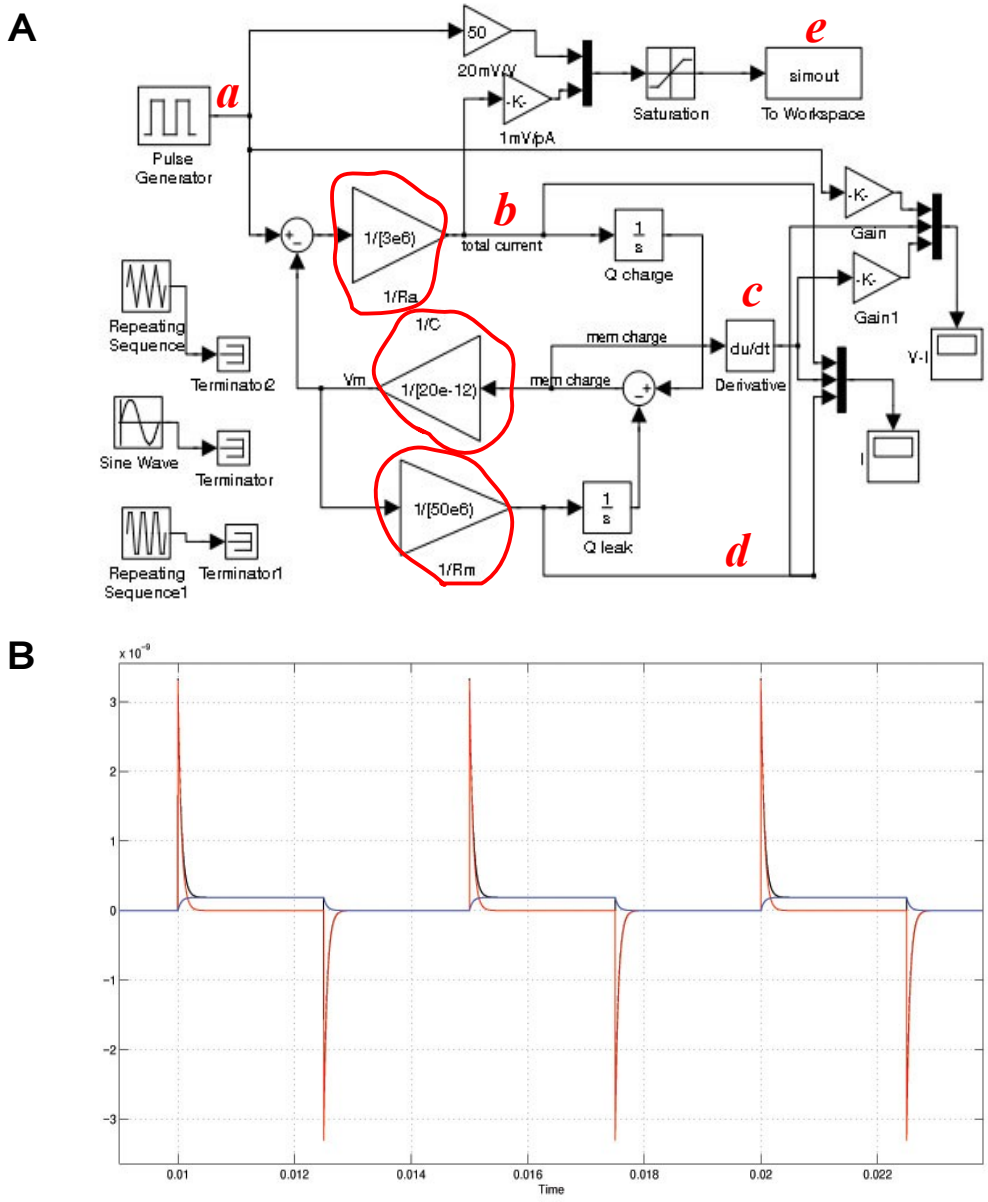


Figure 4.9. Generation of the model current using Simulink. Model diagram of the whole-cell patch clamping configuration is shown in **A**. Cell parameters can be adjusted in the model (red circles). The wave protocol (square pulses in this case, a in **A**) drives the circuit, and the total current (b in **A**, black in **B**), current charging the membrane (c in **A**, red in **B**), and current leaking through the membrane (d in **A**, blue in **B**) are displayed. The sampled data are exported (e in **A**) to the Workspace of MATLAB for analysis.

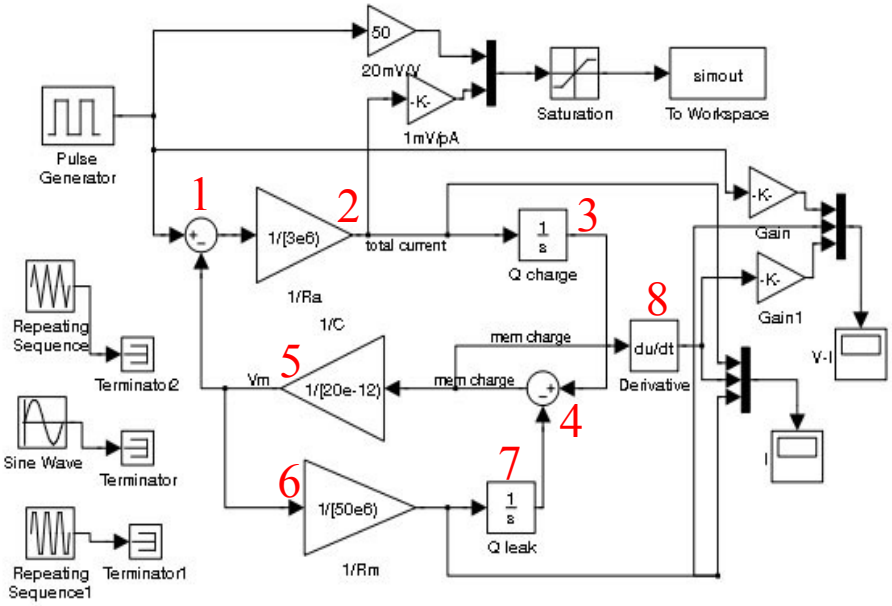


Figure 4.10. Some more about the diagram in Simulink. To explain a little bit more about how the model works, let's follow the numbers in the figure. Total current flowing through the electrode is driven by voltage difference between the electrode and the membrane (*Ra* is in between). In step **1**, membrane potential (from step **5**) is subtracted from command potential, and then divided by *Ra* in step **2** to get the actual current flow. Part of transferred charges (step **3**) are used to charge the membrane (step **4**), and part of them are leaking out of the cell (step **7**). Accumulation of the charges on the membrane builds up membrane potential, and the value is *Q/Cm* (step **5**). As the membrane potential increases, current leaking out of the cell increases, and the value is *Vm/Rm* (step **6**). The leaked charges (step **7**) is also subtracted (step **4**) from the total charges (step **3**). To monitor the net current used to charge the membrane, the derivative of total membrane charges is calculated in step **8**.

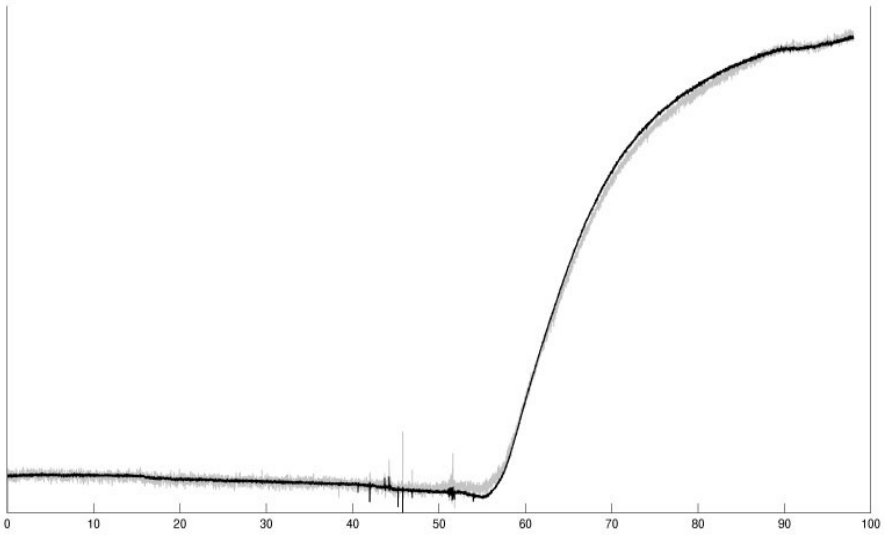


Figure 4.11. Square-wave perturbation – PSD analysis of SQA data. PSD always gives better signal-to-noise ratio compared with SQA. Since square waves are composed of sine waves with different harmonies and amplitudes in the Fourier space, PSD can also be applied when square pulses are given. An example is shown above. The noise of Q-SQA acquired data (gray) is suppressed after PSD analysis (black).

Bibliography

Ali, K., Bilancio, A., Thomas, M., Pearce, W., Gilfillan, A.M. et al. (2004). Essential role for the p110delta phosphoinositide 3-kinase in the allergic response. *Nature*. 431:1007-1011.

Allen, V., Swigart, P., Cheung, R., Cockcroft, S. and Katan, M. (1997). Regulation of inositol lipid-specific phospholipase cdelta by changes in ca2+ ion concentrations. *Biochem J*. 327 (Pt 2):545-552.

Almers, W. and Neher, E. (1987). Gradual and stepwise changes in the membrane capacitance of rat peritoneal mast cells. *J Physiol*. 386:205-217.

Aoyagi, K., Sugaya, T., Umeda, M., Yamamoto, S., Terakawa, S. et al. (2005). The activation of exocytotic sites by the formation of phosphatidylinositol 4,5-bisphosphate microdomains at syntaxin clusters. *J Biol Chem*. 280:17346-17352.

Bai, J., Tucker, W.C. and Chapman, E.R. (2004). Pip2 increases the speed of response of synaptotagmin and steers its membrane-penetration activity toward the plasma membrane. *Nat Struct Mol Biol*. 11:36-44.

Balla, A., Kim, Y.J., Varnai, P., Szentpetery, Z., Knight, Z. et al. (2008). Maintenance of hormone-sensitive phosphoinositide pools in the plasma membrane requires phosphatidylinositol 4-kinase iii{alpha} *Mol Biol Cell*. 19:711-721.

Barnett, D.W. and Misler, S. (1997). An optimized approach to membrane capacitance estimation using dual-frequency excitation. *Biophys J*. 72:1641-1658.

Blackwood, R.A., Smolen, J.E., Transue, A., Hessler, R.J., Harsh, D.M. et al. (1997). Phospholipase d activity facilitates ca2+-induced aggregation and fusion of complex liposomes. *Am J Physiol*. 272:C1279-85.

Borgonovo, B., Cocucci, E., Racchetti, G., Podini, P., Bachi, A. et al. (2002). Regulated exocytosis: a novel, widely expressed system. *Nat Cell Biol*. 4:955-962.

Bronk, P., Deák, F., Wilson, M.C., Liu, X., Südhof, T.C. et al. (2007). Differential effects of snap-25 deletion on ca2+ -dependent and ca2+ -independent neurotransmission. *J Neurophysiol*. 98:794-806.

Brose, N., Petrenko, A.G., Südhof, T.C. and Jahn, R. (1992). Synaptotagmin: a calcium

sensor on the synaptic vesicle surface. *Science*. 256:1021-1025.

Brown, W.J., Chambers, K. and Doody, A. (2003). Phospholipase a2 (pla2) enzymes in membrane trafficking: mediators of membrane shape and function. *Traffic*. 4:214-221.

Burgoyne, R.D. and Morgan, A. (2003). Secretory granule exocytosis. *Physiol Rev.* 83:581-632.

Carr, C.M., Grote, E., Munson, M., Hughson, F.M. and Novick, P.J. (1999). Sec1p binds to snare complexes and concentrates at sites of secretion. *J Cell Biol*. 146:333-344.

Caumont, A.S., Galas, M.C., Vitale, N., Aunis, D. and Bader, M.F. (1998). Regulated exocytosis in chromaffin cells. translocation of arf6 stimulates a plasma membrane-associated phospholipase d. *J Biol Chem*. 273:1373-1379.

Caumont, A.S., Vitale, N., Gensse, M., Galas, M.C., Casanova, J.E. et al. (2000). Identification of a plasma membrane-associated guanine nucleotide exchange factor for arf6 in chromaffin cells. possible role in the regulated exocytotic pathway. *J Biol Chem*. 275:15637-15644.

Chakrabarti, S., Kobayashi, K.S., Flavell, R.A., Marks, C.B., Miyake, K. et al. (2003). Impaired membrane resealing and autoimmune myositis in synaptotagmin vii-deficient mice. *J Cell Biol*. 162:543-549.

Chen, X., Araç, D., Wang, T.M., Gilpin, C.J., Zimmerberg, J. et al. (2006). Snare-mediated lipid mixing depends on the physical state of the vesicles. *Biophys J*. 90:2062-2074.

Chen, Y.A., Scales, S.J. and Scheller, R.H. (2001). Sequential snare assembly underlies priming and triggering of exocytosis. *Neuron*. 30:161-170.

Chernomordik, L.V., Vogel, S.S., Sokoloff, A., Onaran, H.O., Leikina, E.A. et al. (1993). Lysolipids reversibly inhibit ca(2+)-, gtp- and ph-dependent fusion of biological membranes. *FEBS Lett*. 318:71-76.

Choi, W.S., Kim, Y.M., Combs, C., Frohman, M.A. and Beaven, M.A. (2002). Phospholipases d1 and d2 regulate different phases of exocytosis in mast cells. *J Immunol*. 168:5682-5689.

Chow, R.H. and von Rüden Ludolf Electrochemical detection of secretion from single

cells. In *Single-channel recording*. Sakmann, B. and Neher, E. 1995. 245-275.

Clifford, E.E., Parker, K., Humphreys, B.D., Kertesy, S.B. and Dubyak, G.R. (1998). The p2x1 receptor, an adenosine triphosphate-gated cation channel, is expressed in human platelets but not in human blood leukocytes. *Blood*. 91:3172-3181.

Cohen, J.S. and Brown, H.A. (2001). Phospholipases stimulate secretion in rbl mast cells. *Biochemistry*. 40:6589-6597.

Cool, D.E. and Blum, J.J. (1993). Protein tyrosine phosphatase activity in leishmania donovani. *Mol Cell Biochem*. 127-128:143-149.

Coorssen, J.R., Schmitt, H. and Almers, W. (1996). Ca2+ triggers massive exocytosis in chinese hamster ovary cells. *EMBO J*. 15:3787-3791.

Cousin, M.A., Malladi, C.S., Tan, T.C., Raymond, C.R., Smillie, K.J. et al. (2003). Synapsin i-associated phosphatidylinositol 3-kinase mediates synaptic vesicle delivery to the readily releasable pool. *J Biol Chem*. 278:29065-29071.

De Matteis, M.A. and Godi, A. (2004). Pi-loting membrane traffic. *Nat Cell Biol*. 6:487-492.

Dennis, E.A. (1994). Diversity of group types, regulation, and function of phospholipase a2. *J Biol Chem*. 269:13057-13060.

Dernick, G., Alvarez de Toledo, G. and Lindau, M. (2003). Exocytosis of single chromaffin granules in cell-free inside-out membrane patches. *Nat Cell Biol*. 5:358-362.

Dernick, G., Gong, L., Tabares, L., Alvarez de Toledo, G. and Lindau, M. (2005). Patch amperometry: high-resolution measurements of single-vesicle fusion and release. *Nat Methods*. 2:699-708.

Dougherty, P.J., Davis, M.J., Zawieja, D.C. and Muthuchamy, M. (2008). Calcium sensitivity and cooperativity of permeabilized rat mesenteric lymphatics. *Am J Physiol Regul Integr Comp Physiol*. :.

Dulubova, I., Khvotchev, M., Liu, S., Huryeva, I., Südhof, T.C. et al. (2007). Munc18-1 binds directly to the neuronal snare complex. *Proc Natl Acad Sci U S A*. 104:2697-2702.

Dulubova, I., Sugita, S., Hill, S., Hosaka, M., Fernandez, I. et al. (1999). A conformational switch in syntaxin during exocytosis: role of munc18. *EMBO J*. 18:4372-4382.

Eberhard, D.A., Cooper, C.L., Low, M.G. and Holz, R.W. (1990). Evidence that the inositol phospholipids are necessary for exocytosis. loss of inositol phospholipids and inhibition of secretion in permeabilized cells caused by a bacterial phospholipase c and removal of atp. *Biochem J*. 268:15-25.

Fan, J.S. and Palade, P. (1998). Perforated patch recording with beta-escin. *Pflugers Arch*. 436:1021-1023.

Fanara, P., Hodel, M.R., Corbett, A.H. and Hodel, A.E. (2000). Quantitative analysis of nuclear localization signal (nls)-importin alpha interaction through fluorescence depolarization. evidence for auto-inhibitory regulation of nls binding. *J Biol Chem*. 275:21218-21223.

Ferby, I.M., Waga, I., Hoshino, M., Kume, K. and Shimizu, T. (1996). Wortmannin inhibits mitogen-activated protein kinase activation by platelet-activating factor through a mechanism independent of p85/p110-type phosphatidylinositol 3-kinase. *J Biol Chem*. 271:11684-11688.

Fernández-Chacón, R., Königstorfer, A., Gerber, S.H., García, J., Matos, M.F. et al. (2001). Synaptotagmin i functions as a calcium regulator of release probability. *Nature*. 410:41-49.

Fernández-Chacón, R., Shin, O., Königstorfer, A., Matos, M.F., Meyer, A.C. et al. (2002). Structure/function analysis of ca2+ binding to the c2a domain of synaptotagmin 1. *J Neurosci*. 22:8438-8446.

Fix, M., Melia, T.J., Jaiswal, J.K., Rappoport, J.Z., You, D. et al. (2004). Imaging single membrane fusion events mediated by snare proteins. *Proc Natl Acad Sci U S A*. 101:7311-7316.

Fukami, K., Nakao, K., Inoue, T., Kataoka, Y., Kurokawa, M. et al. (2001). Requirement of phospholipase cdelta4 for the zona pellucida-induced acrosome reaction. *Science*. 292:920-923.

Galetic, I., Andjelkovic, M., Meier, R., Brodbeck, D., Park, J. et al. (1999). Mechanism of protein kinase b activation by insulin/insulin-like growth factor-1 revealed by specific inhibitors of phosphoinositide 3-kinase--significance for diabetes and cancer. *Pharmacol Ther.* 82:409-425.

Geppert, M., Goda, Y., Hammer, R.E., Li, C., Rosahl, T.W. et al. (1994). Synaptotagmin i: a major ca2+ sensor for transmitter release at a central synapse. *Cell*. 79:717-727.

Gil, A., Viniegra, S. and Gutiérrez, L.M. (2001). Temperature and pma affect different phases of exocytosis in bovine chromaffin cells. *Eur J Neurosci*. 13:1380-1386.

Goñi, F.M. and Alonso, A. (1999). Structure and functional properties of diacylglycerols in membranes. *Prog Lipid Res*. 38:1-48.

Grishanin, R.N., Kowalchyk, J.A., Klenchin, V.A., Ann, K., Earles, C.A. et al. (2004). Caps acts at a prefusion step in dense-core vesicle exocytosis as a pip2 binding protein. *Neuron*. 43:551-562.

Hilgemann, D.W. and Lu, C.C. (1998). Giant membrane patches: improvements and applications. *Methods Enzymol*. 293:267-280.

Hu, C., Ahmed, M., Melia, T.J., Söllner, T.H., Mayer, T. et al. (2003). Fusion of cells by flipped snares. *Science*. 300:1745-1749.

Huang, C.L., Feng, S. and Hilgemann, D.W. (1998). Direct activation of inward rectifier potassium channels by pip2 and its stabilization by gbetagamma. *Nature*. 391:803-806.

Jahn, R., Lang, T. and Südhof, T.C. (2003). Membrane fusion. *Cell*. 112:519-533.

Jaiswal, J.K., Andrews, N.W. and Simon, S.M. (2002). Membrane proximal lysosomes are the major vesicles responsible for calcium-dependent exocytosis in nonsecretory cells. *J Cell Biol*. 159:625-635.

Jun, Y., Fratti, R.A. and Wickner, W. (2004). Diacylglycerol and its formation by phospholipase c regulate rab- and snare-dependent yeast vacuole fusion. *J Biol Chem*. 279:53186-53195.

Kanemaru, K., Okubo, Y., Hirose, K. and Iino, M. (2007). Regulation of neurite growth by spontaneous ca2+ oscillations in astrocytes. *J Neurosci*. 27:8957-8966.

Karli, U.O., Schäfer, T. and Burger, M.M. (1990). Fusion of neurotransmitter vesicles with target membrane is calcium independent in a cell-free system. *Proc Natl Acad Sci U S A*. 87:5912-5915.

Khvotchev, M., Dulubova, I., Sun, J., Dai, H., Rizo, J. et al. (2007). Dual modes of munc18-1/snare interactions are coupled by functionally critical binding to syntaxin-1 n terminus. *J Neurosci*. 27:12147-12155.

Koushika, S.P., Richmond, J.E., Hadwiger, G., Weimer, R.M., Jorgensen, E.M. et al. (2001). A post-docking role for active zone protein rim. *Nat Neurosci*. 4:997-1005.

Lang, T., Bruns, D., Wenzel, D., Riedel, D., Holroyd, P. et al. (2001). Snares are concentrated in cholesterol-dependent clusters that define docking and fusion sites for exocytosis. *EMBO J*. 20:2202-2213.

Latham, C.F., Osborne, S.L., Cryle, M.J. and Meunier, F.A. (2007). Arachidonic acid potentiates exocytosis and allows neuronal snare complex to interact with munc18a. *J Neurochem*. 100:1543-1554.

Lee, S.B., Várnai, P., Balla, A., Jalink, K., Rhee, S. et al. (2004). The pleckstrin homology domain of phosphoinositide-specific phospholipase cdelta4 is not a critical determinant of the membrane localization of the enzyme. *J Biol Chem*. 279:24362-24371.

Lindau, M. and Neher, E. (1988). Patch-clamp techniques for time-resolved capacitance measurements in single cells. *Pflugers Arch*. 411:137-146.

Lindmo, K. and Stenmark, H. (2006). Regulation of membrane traffic by phosphoinositide 3-kinases. *J Cell Sci*. 119:605-614.

Loyet, K.M., Kowalchyk, J.A., Chaudhary, A., Chen, J., Prestwich, G.D. et al. (1998). Specific binding of phosphatidylinositol 4,5-bisphosphate to calcium-dependent activator protein for secretion (caps), a potential phosphoinositide effector protein for regulated exocytosis. *J Biol Chem*. 273:8337-8343.

MacDonald, P.E., Obermüller, S., Vikman, J., Galvanovskis, J., Rorsman, P. et al. (2005). Regulated exocytosis and kiss-and-run of synaptic-like microvesicles in ins-1 and primary rat beta-cells. *Diabetes*. 54:736-743.

Mahal, L.K., Sequeira, S.M., Gureasko, J.M. and Söllner, T.H. (2002). Calcium-independent stimulation of membrane fusion and snarepin formation by synaptotagmin i. *J Cell Biol*. 158:273-282.

Mahmoud, S.F. and Fewtrell, C. (2001). Microdomains of high calcium are not required for exocytosis in rbl-2h3 mucosal mast cells. *J Cell Biol*. 153:339-349.

Mayer, A., Scheglmann, D., Dove, S., Glatz, A., Wickner, W. et al. (2000). Phosphatidylinositol 4,5-bisphosphate regulates two steps of homotypic vacuole fusion. *Mol Biol Cell*. 11:807-817.

McLaughlin, S. and Murray, D. (2005). Plasma membrane phosphoinositide organization by protein electrostatics. *Nature*. 438:605-611.

McNeil, P.L. and Kirchhausen, T. (2005). An emergency response team for membrane repair. *Nat Rev Mol Cell Biol*. 6:499-505.

Metcalfe, D.D., Baram, D. and Mekori, Y.A. (1997). Mast cells. *Physiol Rev.* 77:1033-1079.

Meunier, F.A., Osborne, S.L., Hammond, G.R.V., Cooke, F.T., Parker, P.J. et al. (2005). Phosphatidylinositol 3-kinase c2alpha is essential for atp-dependent priming of neurosecretory granule exocytosis. *Mol Biol Cell*. 16:4841-4851.

Milosevic, I., Sørensen, J.B., Lang, T., Krauss, M., Nagy, G. et al. (2005). Plasmalemmal phosphatidylinositol-4,5-bisphosphate level regulates the releasable vesicle pool size in chromaffin cells. *J Neurosci*. 25:2557-2565.

Mitchell, C.J., Kelly, M.M., Blewitt, M., Wilson, J.R. and Biden, T.J. (2001). Phospholipase c-gamma mediates the hydrolysis of phosphatidylinositol, but not of phosphatidylinositol 4,5-bisphoshate, in carbamylcholine-stimulated islets of langerhans. *J Biol Chem*. 276:19072-19077.

Mousley, C.J., Tyeryar, K.R., Vincent-Pope, P. and Bankaitis, V.A. (2007). The sec14-superfamily and the regulatory interface between phospholipid metabolism and membrane trafficking. *Biochim Biophys Acta*. 1771:727-736.

Mundroff, M.L. and Wightman, R.M. Amperometry and cyclic voltammetry with carbon fiber microelectrodes at single cells. In *Current protocols in neuroscience*. Gerfen,

C.R., Holmes, A., Rogawski, M.A., Sibley, D., Skolnick, P. and Wray, S. 2002. 6.14.1-6.14.22.

Nagao, T., Kubo, T., Fujimoto, R., Nishio, H., Takeuchi, T. et al. (1995). Ca(2+)-independent fusion of secretory granules with phospholipase a2-treated plasma membranes in vitro. *Biochem J*. 307 (Pt 2):563-569.

Nakamura, Y., Fukami, K., Yu, H., Takenaka, K., Kataoka, Y. et al. (2003). Phospholipase cdelta1 is required for skin stem cell lineage commitment. *EMBO J*. 22:2981-2991.

Nakanishi, S., Kakita, S., Takahashi, I., Kawahara, K., Tsukuda, E. et al. (1992). Wortmannin, a microbial product inhibitor of myosin light chain kinase. *J Biol Chem*. 267:2157-2163.

Nasuhoglu, C., Feng, S., Mao, Y., Shammat, I., Yamamato, M. et al. (2002). Modulation of cardiac pip2 by cardioactive hormones and other physiologically relevant interventions. *Am J Physiol Cell Physiol*. 283:C223-34.

Neher, E. (2006). A comparison between exocytic control mechanisms in adrenal chromaffin cells and a glutamatergic synapse. *Pflugers Arch*. 453:261-268.

Olsen, H.L., Hoy, M., Zhang, W., Bertorello, A.M., Bokvist, K. et al. (2003). Phosphatidylinositol 4-kinase serves as a metabolic sensor and regulates priming of secretory granules in pancreatic beta cells. *Proc Natl Acad Sci U S A*. 100:5187-5192.

Osipchuk, Y. and Cahalan, M. (1992). Cell-to-cell spread of calcium signals mediated by atp receptors in mast cells. *Nature*. 359:241-244.

Ostrowicz, C.W., Meiringer, C.T.A. and Ungermann, C. (2008). Yeast vacuole fusion: a model system for eukaryotic endomembrane dynamics. *Autophagy*. 4:5-19.

Pallen, C.J. and Tong, P.H. (1991). Elevation of membrane tyrosine phosphatase activity in density-dependent growth-arrested fibroblasts. *Proc Natl Acad Sci U S A*. 88:6996-7000.

Patton, C., Thompson, S. and Epel, D. (2004). Some precautions in using chelators to buffer metals in biological solutions. *Cell Calcium*. 35:427-431.

Perin, M.S., Fried, V.A., Mignery, G.A., Jahn, R. and Südhof, T.C. (1990). Phospholipid binding by a synaptic vesicle protein homologous to the regulatory region of protein

kinase c. *Nature*. 345:260-263.

Pertile, P., Liscovitch, M., Chalifa, V. and Cantley, L.C. (1995). Phosphatidylinositol 4,5-bisphosphate synthesis is required for activation of phospholipase d in u937 cells. *J Biol Chem*. 270:5130-5135.

Poole, A.R., Howell, J.I. and Lucy, J.A. (1970). Lysolecithin and cell fusion. *Nature*. 227:810-814.

Qin, W., Pappan, K. and Wang, X. (1997). Molecular heterogeneity of phospholipase d (pld). cloning of pldgamma and regulation of plant pldgamma, -beta, and -alpha by polyphosphoinositides and calcium. *J Biol Chem*. 272:28267-28273.

Rhee, J., Li, L.Y., Shin, O., Rah, J., Rizo, J. et al. (2005). Augmenting neurotransmitter release by enhancing the apparent ca2+ affinity of synaptotagmin 1. *Proc Natl Acad Sci U S A*. 102:18664-18669.

Rhee, J.S., Betz, A., Pyott, S., Reim, K., Varoqueaux, F. et al. (2002). Beta phorbol ester- and diacylglycerol-induced augmentation of transmitter release is mediated by munc13s and not by pkcs. *Cell*. 108:121-133.

Rhee, S.G. (2001). Regulation of phosphoinositide-specific phospholipase c. *Annu Rev Biochem*. 70:281-312.

Richmond, J.E., Weimer, R.M. and Jorgensen, E.M. (2001). An open form of syntaxin bypasses the requirement for unc-13 in vesicle priming. *Nature*. 412:338-341.

Rickman, C. and Davletov, B. (2005). Arachidonic acid allows snare complex formation in the presence of munc18. *Chem Biol*. 12:545-553.

Rigaud, J. and Lévy, D. (2003). Reconstitution of membrane proteins into liposomes. *Methods Enzymol*. 372:65-86.

Rigaud, J.L., Pitard, B. and Levy, D. (1995). Reconstitution of membrane proteins into liposomes: application to energy-transducing membrane proteins. *Biochim Biophys Acta*. 1231:223-246.

Rizo, J. (2003). Snare function revisited. *Nat Struct Biol*. 10:417-419.

Rizo, J. and Südhof, T.C. (2002). Snares and munc18 in synaptic vesicle fusion. *Nat Rev Neurosci*. 3:641-653.

Rizo, J., Chen, X. and Araç, D. (2006). Unraveling the mechanisms of synaptotagmin and snare function in neurotransmitter release. *Trends Cell Biol*. 16:339-350.

Sagi-Eisenberg, R. (2007). The mast cell: where endocytosis and regulated exocytosis meet. *Immunol Rev*. 217:292-303.

Sakaba, T., Stein, A., Jahn, R. and Neher, E. (2005). Distinct kinetic changes in neurotransmitter release after snare protein cleavage. *Science*. 309:491-494.

Santos-Sacchi, J. (2004). Determination of cell capacitance using the exact empirical solution of partial differential y/partial differential cm and its phase angle. *Biophys J*. 87:714-727.

Sarantopoulos, C., McCallum, J.B., Kwok, W. and Hogan, Q. (2004). Beta-escin diminishes voltage-gated calcium current rundown in perforated patch-clamp recordings from rat primary afferent neurons. *J Neurosci Methods*. 139:61-68.

Schuette, C.G., Hatsuzawa, K., Margittai, M., Stein, A., Riedel, D. et al. (2004). Determinants of liposome fusion mediated by synaptic snare proteins. *Proc Natl Acad Sci U S A*. 101:2858-2863.

Schwab, R.B., Okamoto, T., Scherer, P.E. and Lisanti, M.P. Analysis of the association of proteins with membranes. In *Current protocols in cell biology*. Bonifacino, J.S., Dasso, M., Harford, J.B., Lippincott-Schwartz, J. and Yamada, K.M. 2000. 5.4.1-5.4.17.

Sigal, C.T., Zhou, W., Buser, C.A., McLaughlin, S. and Resh, M.D. (1994). Amino-terminal basic residues of src mediate membrane binding through electrostatic interaction with acidic phospholipids. *Proc Natl Acad Sci U S A*. 91:12253-12257.

Smith, A.J., Pfeiffer, J.R., Zhang, J., Martinez, A.M., Griffiths, G.M. et al. (2003). Microtubule-dependent transport of secretory vesicles in rbl-2h3 cells. *Traffic*. 4:302-312.

Smith, C., Moser, T., Xu, T. and Neher, E. (1998). Cytosolic ca2+ acts by two separate pathways to modulate the supply of release-competent vesicles in chromaffin cells. *Neuron*. 20:1243-1253.

Spudich, A. and Braunstein, D. (1995). Large secretory structures at the cell surface

imaged with scanning force microscopy. *Proc Natl Acad Sci U S A*. 92:6976-6980.

Südhof, T.C. and Rizo, J. (1996). Synaptotagmins: c2-domain proteins that regulate membrane traffic. *Neuron*. 17:379-388.

Tang, J., Maximov, A., Shin, O., Dai, H., Rizo, J. et al. (2006). A complexin/synaptotagmin 1 switch controls fast synaptic vesicle exocytosis. *Cell*. 126:1175-1187.

Thompson, R.E., Lindau, M. and Webb, W.W. (2001). Robust, high-resolution, whole cell patch-clamp capacitance measurements using square wave stimulation. *Biophys J*. 81:937-948.

Thorngren, N., Collins, K.M., Fratti, R.A., Wickner, W. and Merz, A.J. (2004). A soluble snare drives rapid docking, bypassing atp and sec17/18p for vacuole fusion. *EMBO J*. 23:2765-2776.

Togo, T., Alderton, J.M., Bi, G.Q. and Steinhardt, R.A. (1999). The mechanism of facilitated cell membrane resealing. *J Cell Sci*. 112 (Pt 5):719-731.

Tucker, W.C., Weber, T. and Chapman, E.R. (2004). Reconstitution of ca2+-regulated membrane fusion by synaptotagmin and snares. *Science*. 304:435-438.

Verhage, M., Maia, A.S., Plomp, J.J., Brussaard, A.B., Heeroma, J.H. et al. (2000). Synaptic assembly of the brain in the absence of neurotransmitter secretion. *Science*. 287:864-869.

Vitale, N., Mawet, J., Camonis, J., Regazzi, R., Bader, M. et al. (2005). The small gtpase rala controls exocytosis of large dense core secretory granules by interacting with arf6-dependent phospholipase d1. *J Biol Chem*. 280:29921-29928.

Wang, P., Chicka, M.C., Bhalla, A., Richards, D.A. and Chapman, E.R. (2005). Synaptotagmin vii is targeted to secretory organelles in pc12 cells, where it functions as a high-affinity calcium sensor. *Mol Cell Biol*. 25:8693-8702.

Wang, T.M. and Hilgemann, D.W. (2008). Ca-dependent non-secretory vesicle fusion in a secretory cell. *J Gen Physiol.* **In press**.

Weber, T., Zemelman, B.V., McNew, J.A., Westermann, B., Gmachl, M. et al. (1998). Snarepins: minimal machinery for membrane fusion. *Cell*. 92:759-772.

Whiteheart, S.W., Rossnagel, K., Buhrow, S.A., Brunner, M., Jaenicke, R. et al. (1994). N-ethylmaleimide-sensitive fusion protein: a trimeric atpase whose hydrolysis of atp is required for membrane fusion. *J Cell Biol*. 126:945-954.

Williams, R.M., Shear, J.B., Zipfel, W.R., Maiti, S. and Webb, W.W. (1999). Mucosal mast cell secretion processes imaged using three-photon microscopy of 5-hydroxytryptamine autofluorescence. *Biophys J*. 76:1835-1846.

Xu, J., Tang, K.S., Lu, V.B., Weerasinghe, C.P., Tse, A. et al. (2005). Maintenance of quantal size and immediately releasable granules in rat chromaffin cells by glucocorticoid. *Am J Physiol Cell Physiol*. 289:C1122-33.

Xu, T., Ashery, U., Burgoyne, R.D. and Neher, E. (1999). Early requirement for alpha-snap and nsf in the secretory cascade in chromaffin cells. *EMBO J*. 18:3293-3304.

Xu, T., Binz, T., Niemann, H. and Neher, E. (1998). Multiple kinetic components of exocytosis distinguished by neurotoxin sensitivity. *Nat Neurosci*. 1:192-200.

Yamaguchi, T., Dulubova, I., Min, S., Chen, X., Rizo, J. et al. (2002). Sly1 binds to golgi and er syntaxins via a conserved n-terminal peptide motif. *Dev Cell*. 2:295-305.

Yaradanakul, A., Wang, T.M., Lariccia, V., Lin, M.J., Shen, C. et al. (2008). Massive Ca-induced membrane fusion and phospholipid changes triggered by reverse Na/Ca exchange in BHK fibroblasts. *J Gen Physiol*. **In press**.

Yoon, T., Okumus, B., Zhang, F., Shin, Y. and Ha, T. (2006). Multiple intermediates in snare-induced membrane fusion. *Proc Natl Acad Sci U S A*. 103:19731-19736.