Quantification of ^{99m}TC-Duramycin Uptake Kinetics in the Area-at-Risk After Myocardial Ischemia and Reperfusion

Joseph Capacete
Marquette University

QUANTIFICATION OF ^{99m}TC-DURAMYCIN UPTAKE KINETICS IN THE AREA-AT-RISK AFTER MYOCARDIAL ISCHEMIA AND REPERFUSION

by

Joseph Capacete

A Thesis submitted to the Faculty of the Graduate School,
Marquette University,
in Partial Fulfillment of the Requirements for
the Degree of Master of Science

Milwaukee, Wisconsin

August 2010

ABSTRACT

Myocardial infarction (MI) is a condition in which blood supply to the heart is insufficient. MI is associated with two forms of cell death: apoptosis and necrosis. ^{99m}Tc-duramycin (^{99m}Tc-D) is a novel radiopharmaceutical that detects cell death by recognizing externalized phosphatidylethanolamine. The objective of this study was to develop a compartmental model for ^{99m}Tc-D uptake kinetics in normal and infarct myocardium, and utilize this model to compare the uptake kinetics of ^{99m}Tc-D in MI with that of another radiopharmaceutical, ^{99m}Tc-C2A-GST.

MI was induced in rats which were then injected (i.v.) with ^{99m}Tc-D. Rats were sacrificed at 3, 10, 20, 60, and 180 minutes after injection. Normal and infarct myocardial tissue were weighed and measured for tracer uptake by gamma counting. ^{99m}Tc-D blood clearance data was also recorded by drawing blood samples at different time points after injection. Results show that ^{99m}Tc-D is sequestered in MI, but not in normal myocardium, that the uptake of ^{99m}Tc-D in MI is 48% higher than for ^{99m}Tc-C2A-GST, and that the blood clearance rate for ^{99m}Tc-D is faster than for ^{99m}Tc-C2A-GST.

We evaluated the ability of three compartmental models, with different assumptions regarding ^{99m}Tc-D uptake kinetics in MI, to interpret the uptake data of ^{99m}Tc-D in normal and infarct myocardium. Model 2, which assumes saturable intracellular probe binding kinetics, and Model 3, which assumes saturable vascular and intracellular probe binding kinetics, provided reasonable fits to ^{99m}Tc-D uptake data that were not different. Both models suggested a ~3-fold larger number of available binding sites for ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST. This is consistent with the fact that phosphatidylethanolamine (^{99m}Tc-D binding target) accounts for ~20% of all

phospholipids in mammalian cellular membranes as compared to ~7% for phosphatidylethanolamine and phosphatidylserine (^{99m}Tc-C2A-GST binding target) combined. Thus, ^{99m}Tc-D and ^{99m}Tc-C2A-GST appear to detect a consistent level of cell death in the same MI model. Models 2 and 3 differed in terms of the contribution of the smaller molecular weight of ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST to ^{99m}Tc-D greater uptake in MI. The results of this study provide a basis for quantitative interpretation of ^{99m}Tc-D *in vivo* uptake in MI.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I
BACKGROUND

1.1 Myocardial Infarct (MI) and Cell Death:

The entire myocardial blood supply is delivered by the left and right coronary arteries which branch from the aorta [1]. The left coronary artery divides into the anterior descending and circumflex arteries, which then supply mainly the left atrium and the anterior and left lateral sections of the left ventricle. The right coronary artery supplies blood to the right atrium and right ventricle as well as the posterior section of the left ventricle in most people [1]. Most of the myocardium is supplied by the coronary arteries and only the inner 1/10 millimeter of the endocardial surface can obtain nutritional blood flow directly from the cardiac chambers [2].

Coronary artery disease (CAD) causes over 500,000 deaths per year and is the leading cause of death in both men and women in the United States [3]. Myocardial infarction (MI) is the most common expression of CAD. MI is a condition in which blood supply to the heart is insufficient, commonly due to an atherosclerotic blockage in a coronary artery. MI is associated with two forms of cell death: apoptosis and necrosis [4]. Apoptosis, or programmed cell death, is a controlled and regulated form of cell death associated with cellular shrinkage and maintenance of plasma membrane integrity [4]. Necrosis on the other hand is associated with cellular swelling, plasma membrane rupture, and an inflammatory response due to the release of intracellular constituents to the environment [4]. Cell death can be difficult to detect in healthy organisms due to efficient clearance of dying cells by phagocytes. In pathologies such as MI, apoptotic and necrotic cells are more easily detected due to the inability of phagocytes to clear the increased number of dying cells [5].

The ability to detect apoptotic and necrotic cells can be useful in identifying infarct tissue for the diagnosis of ischemic heart disease. While apoptosis and necrosis are very different processes, both lead to the exposure of inner membrane phospholipids [6]. Phospholipids such as phosphatidylethanolamine (PE) and phosphatidylserine (PS) are confined to the inner leaflet of the plasma membrane in viable cells and therefore serve as an advantageous molecular signature for cell death in MI. During cell death, aminophospholipid translocase and a scramblase are activated which results in the exposure of previously bound membrane phospholipids [6]. The exposed phospholipids act as markers to be recognized by macrophages for the removal of apoptotic cells. Fadok et al showed that PS was expressed on the surface of apoptotic cells, and confirmed using lymphocytes that PS is exposed on the surface of apoptotic cells as a signal for cell removal to be recognized by macrophages [7]. In another study, Emoto et al demonstrated the redistribution of PE to the cell surface in apoptosis [8]. Molecular imaging techniques have been explored to take advantage of this molecular signature of cell death [5, 9].

The diagnosis of coronary heart disease is currently supported by three key criteria [3]. Physicians examine a patient for a history of chest pain, abnormalities in ECG readings, and certain changes in biomarkers. All three of these criteria are needed for diagnosis and a single quantitative measure to diagnose coronary heart disease is not available. Molecular imaging has the potential to fill this void by using a tracer that can detect infarct myocardium by binding to exposed membrane phospholipids. Imaging of MI is useful for distinguishing between the infarct myocardium itself and the salvageable ischemic tissue that surrounds it.

1.2 Molecular Probes for Imaging MI:

Molecular imaging has been suggested as a comprehensive approach for detecting MI [5, 9]. The exposure of membrane phospholipids on the surface of apoptotic and necrotic cells has been utilized for molecular imaging techniques. The exposed phospholipids provide a binding site for molecular imaging which is a unique molecular signature of apoptotic and necrotic cells. The exploration of the uptake kinetics of various molecular probes is aimed at achieving earlier imaging of MI with a higher signal to background ratio. Previous studies have been performed to investigate potential tracers for applications in diagnosing CAD by detecting MI [5, 9]. Most molecular imaging studies for MI have been conducted using ^{99m}Technetium (^{99m}Tc) labeled radiophamaceuticals. ^{99m}Tc is a metastable nuclear isomer of technetium that has a half-life of about 6 hours [10]. ^{99m}Tc decays by emitting gamma rays at 140 keV and is readily available from ^{99}Mo- ^{99m}Tc generator systems [11].

1.2.1 Annexin V:

The first molecular probe developed to take advantage of the above molecular signature of cell death was annexin V [5]. Annexin V was first discovered as a vascular anticoagulant protein present in the blood vessels. The anticoagulant mechanism of annexin V is based on its ability to bind to PS, which is exposed to the surface of the plasma membrane during cell death. Koopman et al used fluoroscein isothiocynate (FITC) labeled annexin V to detect apoptotic B cells [12]. Kietselaer et al used ^{99m}Tc-labeled annexin V imaging to detect apoptotic cells in the heart during heart failure [13]. The radiolabeled annexin V was found to be taken up early and late in the infarct region. However, annexin V has limitations as a molecular imaging probe due to its large molecular weight (33-36 kDa) that presents a slow blood clearance. A slow blood

clearance rate is unfavorable for the enhancement of the early contrast between target tissue and background [14]. The utility of annexin V is also limited by the relatively high concentration of intracellular endogenous annexin V in cardiomyocytes which competes with the radiolabeled exogenous annexin V for phospholipid binding sites [15].

1.2.2 C2A-GST (Glutathione S-Transferase):

Another molecular probe that has been considered for detection of apoptotic cells is the C2A domain of synaptotagmin I, a synaptic vesicle protein [16]. A 2006 study by Zhao et al examined the use of the C2A-GST (glutathione S-transferase) form of the protein to detect MI in rats [17]. C2A-GST binds to PS, like annexin V, and to phosphatidylinositide (PI) to detect apoptotic cells. C2A-GST is comparable in size to annexin V with a molecular weight of ~ 37 kDa. C2A-GST was labeled with ^{99m}Tc and injected into rats with induced MI as well as healthy rats [17]. Results showed that the uptake of ^{99m}Tc-C2A-GST in myocardial infarct was due to specific binding to exposed membrane phospholipids as well as passive diffusion into infarct tissue [17]. An inactive form of ^{99m}Tc-C2A-GST (^{99m}Tc-C2A-GST-NHS) was developed as a control for infarct myocardium uptake experiments. The inactive form has a similar molecular weight but does not bind to PS or PI binding sites [17].

A compartmental model was developed by Audi et al to quantify the uptake of ^{99m}Tc-C2A-GST in normal and infarct rat myocardium [18]. The model consisted of two physical spaces: vascular and tissue spaces which represented the volume of coronary circulation and the volume of myocardial tissue accessible to ^{99m}Tc-C2A-GST respectively. The model allowed for sequestration of the probe in the vascular and/or tissue spaces due to specific binding in each region. The model consisted of six kinetic model parameters: V_1 (ml), the volume of the vascular space; V_2 (ml), the volume of the

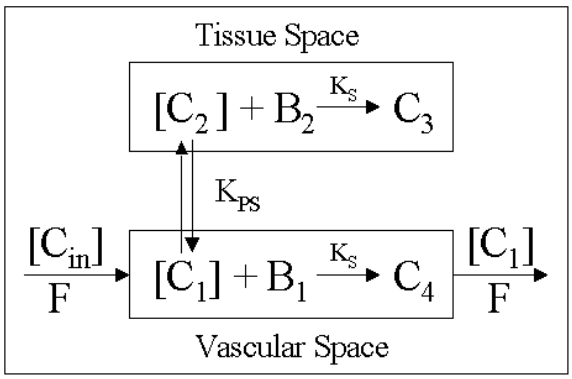
tissue space; K_{PS} (ml/min), the permeability-surface area product which represents the rate of diffusion between the two spaces; K_S (ml/nmol/min), the binding rate of the probe to receptors; B_{01} (nmol), the total amount of specific binding sites available in the vascular space; and B_{02} (nmol), the total amount of specific binding sites available in the tissue space. For the inactive probe, the model assumed that $K_S = 0$ since the inactive probe is unable to bind to the specific binding sites. Using mass balance and mass action, the uptake kinetics of the molecular probe were described by the following differential equations:

$$\frac{d[C_1]}{dt} = \frac{F}{V_1} * ([C_{in}] - [C_1]) + \frac{K_{PS}}{V_1} * ([C_2] - [C_1]) - \frac{K_S}{V_1}[C_1]\,(B_{01} * m - C_4) \quad (1.1)$$

$$\frac{d[C_2]}{dt} = \frac{K_S}{V_2} * ([C_1] - [C_2]) - \frac{K_S}{V_2}[C_2]\,(B_{02} * m - C_3) \quad (1.2)$$

$$\frac{dC_3}{dt} = K_S * m * [C_2]\,(B_{02} * m - C_3) \quad (1.3)$$

$$\frac{dC_4}{dt} = K_S * m * [C_1]\,(B_{01} * m - C_4) \quad (1.4)$$



$[C_{in}]$ = input probe concentration
$[C_1]$ = unbound probe concentration in vascular space
$[C_2]$ = unbound probe concentration in tissue space
C_3 = amount of probe bound in the tissue space
C_4 = amount of probe bound in the vascular space
F = Flow (1 ml/min/g)

Figure 1.1 Compartmental model for the uptake of ^{99m}Tc-C2A-GST in normal and infarct myocardium. K_S is the binding rate constant of the probe; K_{PS} is the rate of diffusion between the tissue and vascular spaces; B_1 and B_2 are the numbers of available binding sites in the vascular and tissue spaces, respectively [18].

For ^{99m}Tc-C2A-GST, Audi et al showed that a combination of saturable cell surface (vascular) and intracellular binding was needed to explain the uptake kinetics of ^{99m}Tc-C2A-GST and its inactive form in rat MI [18].

1.2.3 Duramycin:

Duramycin, a 19-amino acid peptide produced by *Streptoverticillium cinnamoneus*, ^{99m}Tc-duramycin (^{99m}Tc-D) is a novel radiopharmaceutical that detects apoptosis and necrosis by recognizing externalized PE. PE accounts for ~ 20% of all phospholipids in mammalian cellular membranes and therefore provides a greater number of binding sites for molecular imaging than PS and PI combined, which account for only ~7% of phospholipids [19, 9]. While PS serves as a signaling mechanism for the phagocytosis of dying cells, the exposure of PE plays a regulatory role in blebbing during apoptosis by mediating the reorganization of actin filaments. Duramycin is much smaller in size (MW ~ 3 kDa) compared to annexin V (33-36 kDa) and C2A-GST (37 kDa) [14]. Moreover, duramycin is the smallest known polypeptide with a well-defined three-dimensional binding site. In contrast to annexin V and C2A-GST, the binding of duramycin to membrane phospholipids is a calcium-independent process such that duramycin uptake is not limited by the availability of calcium. Duramycin has also been shown to have low uptake in the liver as compared to annexin V, reducing the interference in images from the hepatic background. Thus, ^{99m}Tc-D appears promising for cardiac applications, due to its low molecular weight, fast blood clearance, avid infarct binding and minimal hepatic background [14].

Duramycin is closely related to another 19-amino-acid tetracyclic polypeptide referred to as cinnamycin. The study of cinnamycin has been used to confirm the binding activity of duramycin. Both duramycin and cinnamycin belong to a group of compounds

called lantibiotics. Lantibiotics are defined as bacteriocins that are characterized by a high proportion of unusual amino acids. Cinnamycin was shown to bind to PE on the surface of apoptotic cells in a study by Emoto et al [8]. In that study, cinnamycin was used to provide the first direct evidence of PE translocation to the cell surface during apoptosis. The exposure of PE was compared to the exposure of PS during apoptosis as revealed in previous studies [7, 20]. The time-dependent binding profile of cinnamycin to PE was shown to correlate well with the binding of annexin V to PS, confirming the ability of cinnamycin to detect PE as an indicator of apoptosis [8].

The binding activity of duramycin was compared to that of cinnamycin in order to examine the binding specificity of duramycin [21]. In a study by Iwamoto et al, model membranes were used to investigate the binding of lantibiotics to PE on the surface of cell membranes. The results showed preferential binding of both duramycin and cinnamycin to PE in highly curved membranes. The study by Iwamoto et al confirmed the suggestion that duramycin binds specifically to PE to indicate apoptosis in a manner similar to cinnamycin.

A 2008 study by Zhao et al demonstrated the use of ^{99m}Tc-D for noninvasive imaging of PE [14]. In that study, duramycin was radiolabeled with ^{99m}Tc and studied using binding assays in vitro, whole-body biodistribution in healthy rats, and imaging of acute cardiac cell death in vivo using a rat model of ischemia and reperfusion. In vitro binding tests showed a significant enhancement of ^{99m}Tc-D binding in apoptotic cells versus viable control cells. The cellular binding was also shown to be competitively abolished in the presence of PE-containing liposomes. Briefly, ^{99m}Tc-D was incubated with 2×10^6 viable or apoptotic lymphocytes. The radioactivity in the viable cells was

shown to remain near background while the radioactivity in apoptotic cells was dramatically increased compared to background.

^{99m}Tc-D was also shown to have a rapid blood clearance with a blood half-life of less than 4 min as compared to ~10 min for ^{99m}Tc-C2A-GST. In addition, greater than 95% of the blood activity was shown to be associated with the plasma, confirming the minimal interaction of ^{99m}Tc-D with normal blood cells. The biodistribution study showed that renal excretion is the predominant route of clearance with low uptake in hepatic and gastrointestinal regions. Low uptake in the brain confirmed that ^{99m}Tc-D does not cross the blood-brain barrier. Early and clear detection of MI was enabled by a rapid return to background levels of radioactivity in the lungs and normal myocardium. The biodistribution profile was verified *in vivo* using dynamic imaging analysis.

A focal region of uptake in MI tissue was observed in dynamic images in vivo as early as 10 min after the injection of ^{99m}Tc-D. Preliminary results show that ^{99m}Tc-D has a higher uptake in MI as compared to ^{99m}Tc-C2A-GST. One of the questions addressed in this study is how much of this increase in uptake is due to the increase in the binding kinetics (binding rate constant and/or number of available binding sites) for ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST, and how much is due to the fact that ^{99m}Tc-D has a molecular weight (3 kDa) that is significantly smaller than that of ^{99m}Tc-C2A-GST (37 kDa) which would be expected to increase the tissue permeability of ^{99m}Tc-D as compared to that for ^{99m}Tc-C2A-GST [14].

A partially inactivated form of ^{99m}Tc-D (^{99m}Tc-D^{I}) was developed as a control peptide, which has diminished binding to PE with a minimally altered structure and identical molecular weight as compared to ^{99m}Tc-D [14]. ^{99m}Tc-D^{I} was developed by

modifying a carboxylate group of Asp15 in the binding pocket of duramycin and then purified and labeled in the same manner as the active probe, ^{99m}Tc-D.

1.3 Objective and Specific Aims:

The objective of this thesis is to establish a quantitative basis for interpretation of ^{99m}Tc-Duramycin (^{99m}Tc-D) *in vivo* uptake in normal and infarct myocardial tissue, and for comparison of the myocardial uptake kinetics of ^{99m}Tc-D with other molecular probes of myocardial infarction. The specific aims are 1) to develop a compartmental model for the uptake kinetics of ^{99m}Tc-D in normal and infarct myocardium, and 2) utilize this compartmental model to compare the uptake kinetics of ^{99m}Tc-D with that of another probe ^{99m}Tc-C2A-GST in normal and infarct rat myocardium, and to better understand the underlying mechanisms for the difference in the uptake kinetics of ^{99m}Tc-D and ^{99m}Tc-C2A-GST in rat MI.

CHAPTER II
EXPERIMENTAL METHODS

2.1 Preparation of Active and Partially Inactive ^{99m}Tc-D:

As previously described, duramycin was labeled with ^{99m}Tc by the tricine-phosphine coligand system after being conjugated with HYNIC [14]. The HYNIC-conjugated duramycin was synthesized by reacting HYNIC with duramycin at a molar ratio of 8:1 in the presence of 4 equivalents of triethylamine. 15 μg of the HYNIC-conjugated duramycin was mixed with 40 mg of tricine, 1 mg of trisodium triphenylphosphine-3,3',3"-trisulfonate (TPPTS), and 20 μg, of $SnCl_2$ in 0.8 mL at pH 5.3. The partially inactive form of the probe, ^{99m}Tc-D^{I}, was developed by modifying a carboxylate group of Asp15 in the binding pocket of duramycin and preparing the partially inactive probe in the same manner as the active probe as described above.

2.2 ^{99m}Tc-D Blood Clearance Profile:

^{99m}Tc-D blood clearance in infarcted rats was previously measured. Briefly, blood samples were collected from the contralateral femoral artery in healthy rats at 1, 3, 5, 11, 20, 40, and 60 minutes following the injection of ^{99m}Tc-D [14]. The activity counts in each sample were compared to the counts of the injected dose such that the blood activity was presented as a percentage of the injected dose (%ID).

2.3 Rat Model of Acute Myocardial Infarction (MI):

The rat model of acute myocardial infarction has been previously described [14]. Briefly, each Sprague-Dawley rat was anesthetized with sodium pentobarbital (50 mg/kg) intraperitoneally. After tracheal intubation, respiration was maintained using a rodent respirator. Real-time electrocardiogram (ECG) was monitored throughout the surgery. After thoracotomy, the chest was opened at the fourth intercostal space to expose the heart. MI was induced by ligating the left anterior descending coronary artery using a 6-0

suture at 1 mm below the atrial appendage and reperfusing the artery after 30 minutes. The onset of acute ischemia/reperfusion was confirmed by paleness in the area at risk on the anterior wall of the left ventricle and typical changes in the ECG profile including a significant increase in the QRS profile and immediate elevation of the ST segment. Two hours after initial reperfusion, the rats were injected with ^{99m}Tc-D via the tail vein. The same procedure was performed with ^{99m}Tc-D^{I} injected via the tail vein.

2.4 ^{99m}Tc-D Uptake in Normal (Viable) and Infarct Myocardium:

Rats were sacrificed at 3 (n = 4), 10 (n = 3), 20 (n = 2), 60 (n = 3), and 180 (n = 2) minutes post-injection of ^{99m}Tc-D for a total of fourteen rats. Thirty seconds before sacrifice, the left anterior descending coronary artery was re-occluded by tightening the suture. 2 mL of Evans blue dye was injected into the reoccluded left descending coronary artery before sacrifice and the myocardium devoid of Evans blue was determined to be the area-at-risk for MI (infarct or ischemic tissue). The heart was excised and rinsed in 0.9% saline. The area-at-risk for MI tissue was then dissected from viable myocardium and submerged in pre-warmed 0.5% triphenyltetrazolium chloride (TTC) for 15 minutes at 37°C. The viable tissue (Evans positive, blue) and MI tissue (Evans and TTC negative, pale) was weighed and measured separately for tracer uptake by gamma counting. The activity counts were compared to the counts of the injected dose and corrected for activity decay such that the activity is presented as a percentage of the injected dose per gram of tissue (%ID/g).

CHAPTER III
EXPERIMENTAL RESULTS

3.1 ^{99m}Tc-D Blood Clearance Data:

^{99m}Tc-D blood clearance has an early rapid phase and a late slow phase (Figure 3.1A). About 67% of the injected dose was cleared from the blood 3 minutes post-injection of ^{99m}Tc-D, while about 8% of the injected dose remained in the circulation at 1 hr post-injection. In comparison, C2A-GST blood clearance was significantly slower (Figure 3.1B), with about 49% of the injected dose cleared by 10 minutes post injection, while about 18% of the injected dose remaining in the blood 1 hr post-injection [18]. For subsequent data analysis, the blood clearance profile of ^{99m}Tc-D is assumed to be the same for both active and partially inactive forms since both have similar molecular weights [18].

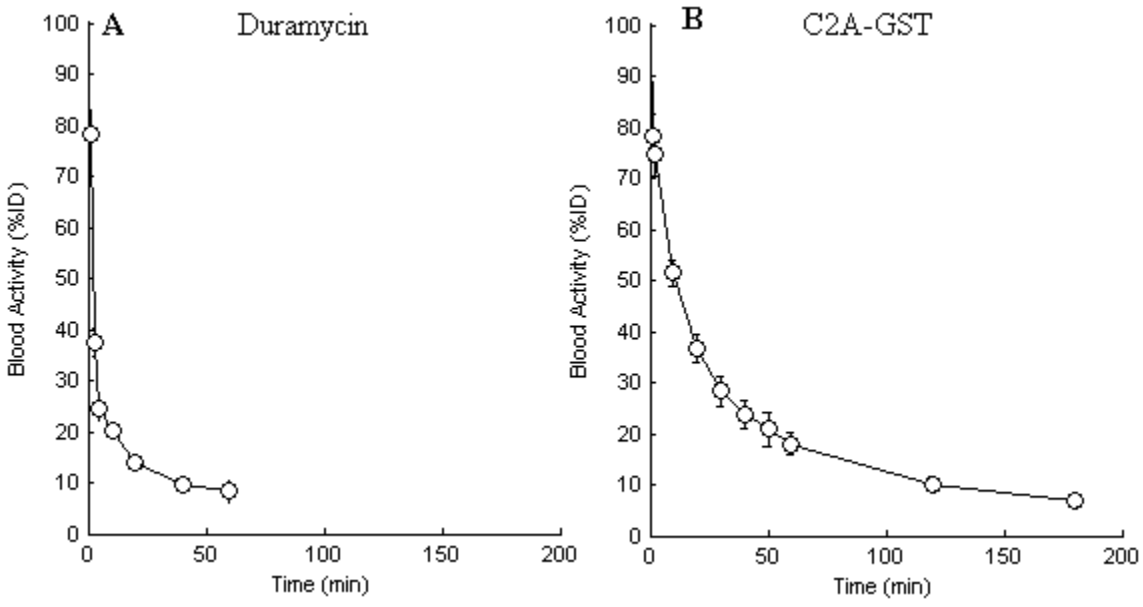


Figure 3.1 A) Blood clearance (in %ID) of ^{99m}Tc-D in healthy rats (mean ± S.D.; n = 2) [14]. B) Blood clearance (in %ID) of ^{99m}Tc-C2A-GST in healthy rats (mean ± S.D.; n = 3) [18].

3.2 ^{99m}Tc-D Uptake in Normal and Infarct Myocardium:

^{99m}Tc-D uptake in normal myocardium (Figure 3.2A) was relatively low (0.27 ± 0.03 (SD) %ID/g at 180 min) presumably due to the inaccessibility of PE binding sites to ^{99m}Tc-D and its fast blood clearance (Figure 3.1A). ^{99m}Tc-D uptake in MI (Figure 3.2A) shows a rapid increase in radioactivity reaching 4.47 ± 0.90 (%ID/g) at 60 min post-injection. This is followed by a slight washout with the activity approaching a plateau of 3.91 ± 0.47 (%ID/g) at 180 minutes post-injection. For the partially inactive form of ^{99m}Tc-D, its uptake in MI shows a slower rise compared to the active form (Figure 3.2A) and a lower plateau at 2.31 (%ID/g) at 180 min post-injection. The uptake of the partially inactive form in normal myocardium is assumed to be the same as that for the active form.

3.3 ^{99m}Tc-C2A-GST Uptake in Normal and Infarct Myocardium:

Previous studies have shown that, like ^{99m}Tc-D, ^{99m}Tc-C2A-GST has low uptake (Figure 3.2B) in normal myocardium (0.11 ± 0.04 (SD) %ID/g at 180 minutes) due to the inaccessibility of ^{99m}Tc-C2A-GST to PS and PI binding sites [17, 18]. The uptake of ^{99m}Tc-C2A-GST in MI had a rapid early phase reaching 2.62 ± 0.92 at 30 min post-injection and stated relatively constant for 180 min (2.63 ± 0.36 (%ID/g)) post-injection. For the inactive form of the probe, early uptake in MI was smaller (peak at 1.06 ± 0.49 %ID/g at 30 minutes) than that for the active form, and decreased with time consistent with the uptake being mostly due to passive diffusion into the tissue instead of specific binding [17, 18].

The uptake of ^{99m}Tc-C2A-GST in MI is significantly lower than that of ^{99m}Tc-D although the amount of probe injected was the same for both probes (~2.3 nmol). Again, one of the questions addressed in this study is how much of this increase in ^{99m}Tc-D uptake in MI is due to the increase in the number of available binding sites for ^{99m}Tc-D

(PE) as compared to ^{99m}Tc-C2A-GST (PS and PI), and how much is due to the low molecular weight of ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST. The latter would suggest an increase in membrane permeability for ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST. Addressing this question is not straightforward since the smaller molecular weight of ^{99m}Tc-D as corresponds to faster blood clearance.

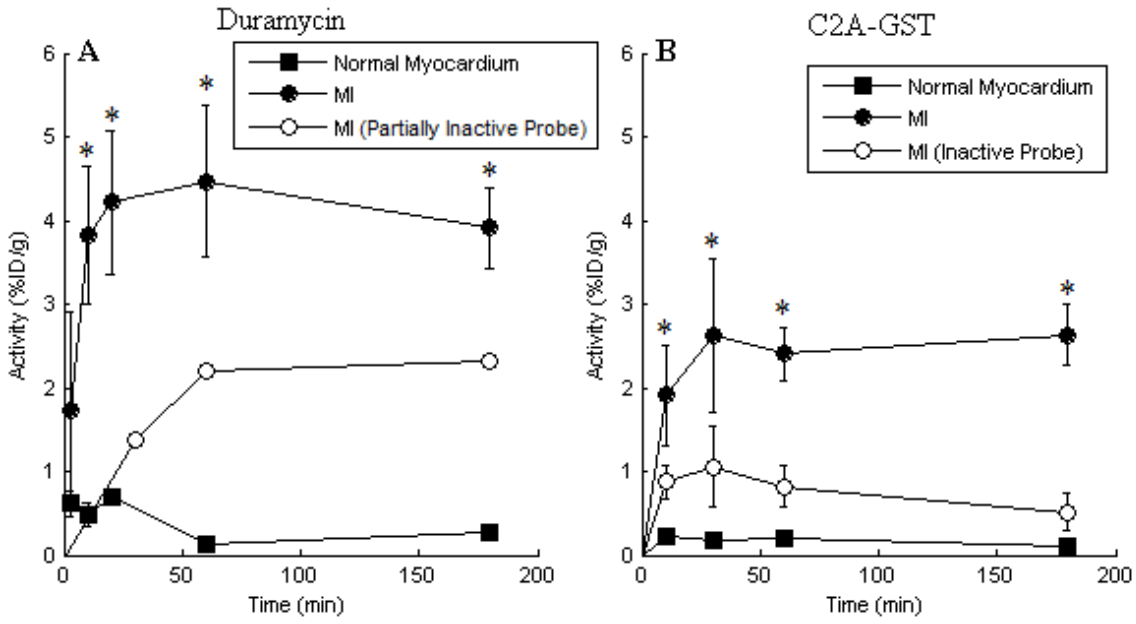


Figure 3.2 A) Radioactivity (in %ID/g) in normal myocardium and MI for active probe at 3 (mean ± S.D.; n = 4), 10 (mean ± S.D.; n = 3), 20 (mean ± S.D.; n = 2), 60 (mean ± S.D.; n = 3), and 180 min (mean ± S.D.; n = 2) after injection. Radioactivity (in %ID/g) in MI for partially inactive probe at 30, 60, and 180 min after injection (n = 1). B) Radioactivity (in %ID/g) in normal myocardium and MI (for active and inactive probe) at 10, 30, 60 and 180 min after injection (mean ± S.D.; n = 4) [17, 18].

CHAPTER IV
DATA ANALYSIS

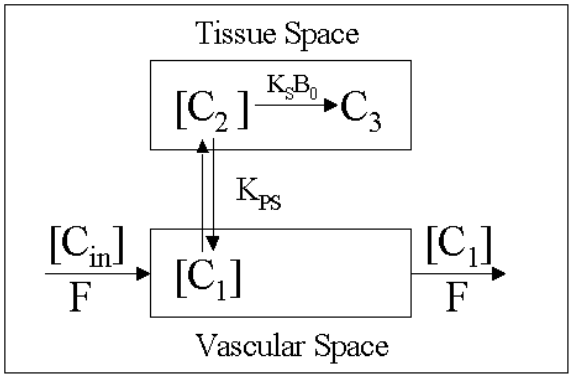
4.1 Motivation for Use of Mathematical Modeling:

The uptake of ^{99m}Tc-D and ^{99m}Tc-C2A-GST and their fully or partially inactive forms in MI is determined by multiple factors, including the number of available binding sites, binding rate constant, accessibility of these binding sites to probe from the vascular region, and probe blood clearance rate. Mathematical modeling can be used to quantify the contribution of each of these factors to the uptake of the probe, and to estimate values of parameters descriptive of these factors. These values will be the basis for determining the relative contribution of each of the above factors to the uptake kinetics of ^{99m}Tc-D and ^{99m}Tc-C2A-GST in MI. Three distinct models were evaluated for their ability to describe the uptake kinetics of ^{99m}Tc-D and ^{99m}Tc-C2A-GST and their fully or partially inactive forms in MI and normal myocardium. What follows is a description of each model and the resulting model fits to the data in Figures 3.1 and 3.2.

4.2 Model 1:

Each of the three models consists of vascular and tissue spaces, with respective volumes V_1 (ml/g) and V_2 (ml/g) with $[C_{in}(t)]$ as the input probe blood concentration curve to the vascular region. The vascular space represents the coronary circulation and the tissue space represents the myocardial tissue accessible to the probe from the coronary circulation.

Figure 4.1 shows a schematic diagram of Model 1 which assumes that the binding of the probe to specific binding sites within the tissue space is unidirectional. Moreover, the model assumes that the number of available PE binding sites is large compared to the number of injected probe molecules available for binding. Thus, the binding of ^{99m}Tc-D to PE within the tissue space is assumed to follow linear kinetics.



$[C_{in}]$ = input probe concentration
$[C_1]$ = unbound probe concentration in vascular space
$[C_2]$ = unbound probe concentration in tissue space
C_3 = amount of probe bound in the tissue space
F = Flow (1 ml/min/g)

Figure 4.1 Schematic diagram of Model 1 for probe uptake in normal and infarct myocardium. K_SB_0 is the rate of probe sequestration within the tissue space; K_{PS} is the rate of probe diffusion between the tissue and vascular spaces.

Using mass balance and mass action, the time course of the activities of the probe in the vascular and tissue spaces are described by Equations (4.1-4.3):

$$\frac{d[C_1]}{dt} = \frac{F}{V_1} * ([C_{in}] - [C_1]) + \frac{K_{PS}}{V_1} * ([C_2] - [C_1]) \quad (4.1)$$

$$\frac{d[C_2]}{dt} = \frac{K_{PS}}{V_2} * ([C_1] - [C_2]) - \frac{K_SB_0}{V_2} * [C_2] \quad (4.2)$$

$$\frac{dC_3}{dt} = K_SB_0 * m * [C_2] \quad (4.3)$$

where $[C_1]$ represents the concentration (μM) of probe present in the vascular space; $[C_2]$ represents the concentration (μM) of unbound probe present in the tissue space; C_3 represents the amount (nmol) of probe sequestered in the tissue space by binding to specific binding sites; m (g) is the mass of the tissue; K_{PS} (ml/min/g) represents the rate of probe diffusion between the vascular and tissue spaces; K_SB_0 (ml/min/g) is the rate of probe sequestration within the tissue region, and is a measure of the total amount of specific binding sites available for probe within the tissue space; and F (ml/min/g) is coronary circulation blood flow which is fixed at 1 ml/min/g [18].

The measured radioactivity from the experimental protocol is the sum of the total radioactivity in the tissue including unbound probe in the vascular space and both bound and unbound probe in the tissue space. $C_{total}(t)$ represents the model total amount of radioactivity in vascular and tissue spaces (in %ID/g) and is given by Equation (4.4):

$$C_{total}(t) = \frac{(V_1 * m * [C_1] + V_2 * m * [C_2] + C_3)(MW * 10^{-9})}{ID * m} * 100 \qquad (4.4)$$

where MW (g/mol) is the molecular weight of the probe and ID (g) is the injected dose. $C_{total}(t)$ is used to compare the model (Equations 4.1-4.3) solution to probe uptake data in myocardium (Figure 3.2).

For Model 1, the unknown kinetic model parameters are V_1, V_2, K_{PS}, and K_SB_0. This model assumes that the active and partially inactive probes have the same V_1, V_2, and K_{PS}, but different K_SB_0 values (K_SB_{0A} and K_SB_{0I} for active and partially inactive probe, respectively).

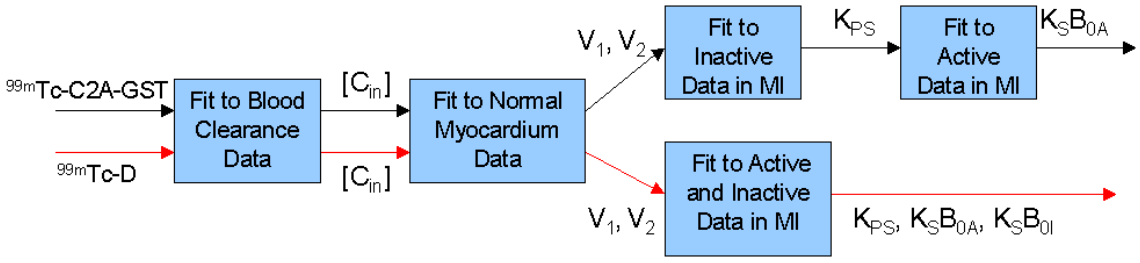


Figure 4.2 Flowchart of the parameter estimation protocol for Model 1. V_1 and V_2 are the volumes of the vascular and tissue spaces, respectively; K_{PS} is the rate of diffusion between the tissue and vascular spaces; K_SB_{0A} and K_SB_{0I} are the rates of tissue sequestration of active and partially inactive probe, respectively.

4.2.1 Estimation of Model 1 Parameters:

For each of the three compartmental models, the MATLAB (The MathWorks, Inc.) function "*ode15s*" was used to obtain a numerical solution of the model's governing

differential equations. "*ode15s*" is a variable order solver which uses numerical differentiation formulas for solving a system of ordinary differential equations [22]. The solution of the differential equations, namely $[C_1](t)$, $[C_2](t)$, and $C_3(t)$ for Model 1, were then substituted into $C_{total}(t)$, which was then fit to probe uptake data. The fitting procedure consisted of finding the values of the model parameters for which $C_{total}(t)$ best fit the probe uptake data (Figure 3.2) in the least-squares sense. This nonlinear parameter optimization procedure was carried out using the MATLAB function '*lsqcurvefit*', which solves a non-linear curve fitting problem in the least-squares sense using the Levenberg-Marquardt algorithm [23].

A flowchart of the parameter estimation protocol for Model 1 is shown in Figure 4.2. The first step was to determine $[C_{in}](t)$, the input function to the vascular space. The following empirical equation was used to describe the blood clearance profile of the probe (Figure 3.1) [18]:

$$C_b(t) = ae^{-\alpha_1 t} + be^{-\alpha_2 t} + \left(100 - a - b\right)e^{-\alpha_3 t} \qquad (4.5)$$

where a, b, and (100 – a – b) are the amplitudes of the three exponential functions with rate constants α_1, α_2, and α_3, respectively. These amplitudes and rate constants are determined by fitting Equation (4.5) to the blood activity data in Figure 3.1. Figure 4.3 shows Equation (4.5) fits to the ^{99m}Tc-D and ^{99m}Tc-C2A-GST blood clearance data.

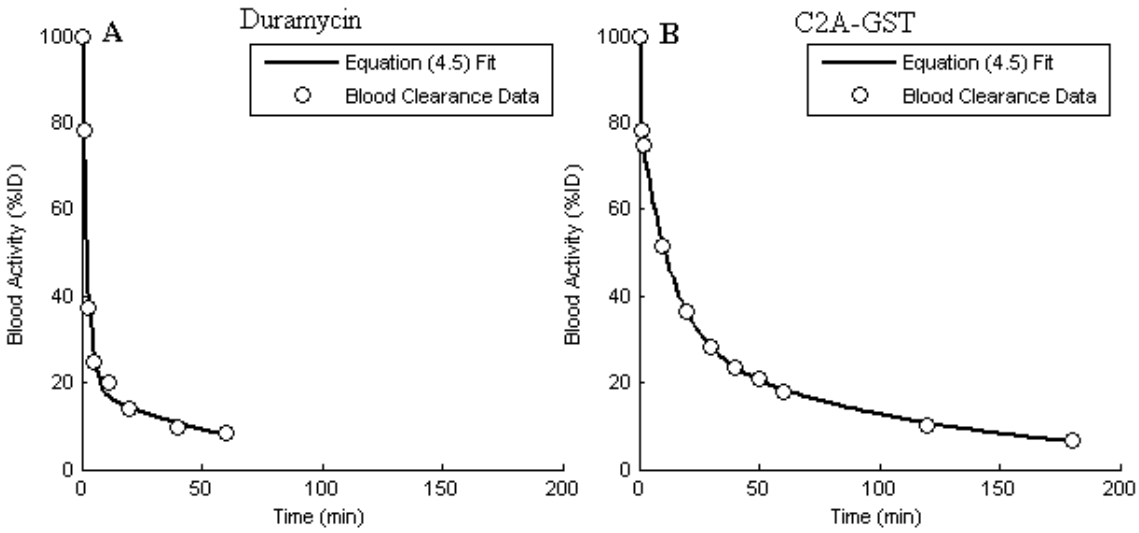


Figure 4.3 A) Blood clearance (in %ID) of ^{99m}Tc-D in healthy rats (mean ± S.D.; n = 2) [14]. B) Blood clearance (in %ID) of ^{99m}Tc-C2A-GST in healthy rats (mean ± S.D.; n = 3) [18]. Solid lines are Equation (4.5) fits.

Knowing $C_b(t)$ in %ID, $[C_{in}](t)$ in μM was determined using Equation (4.6) [18]:

$$[C_{in}] = \frac{\left(C_b\ ID\right) 10^9}{100\ V_{in}\ MW} \tag{4.6}$$

where V_{in} (ml) is the rat blood volume (~ 7% of total body weight) [18].

Since uptake in normal myocardium is relatively low compared to MI for both probes, it is assumed that the probe has no access to the tissue space in normal myocardium. As such, K_{PS} is set to zero and the number of parameters is reduced to one, namely V_1, the volume of the vascular space. Knowing $[C_{in}](t)$, V_1 was estimated by fitting the solution of the governing differential equations with K_{PS} set to zero, expressed as $C_{total}(t)$, to the normal myocardium probe (^{99m}Tc-D or ^{99m}Tc-C2A-GST) uptake data (Figure 3.2). Knowing V_1, V_2 was determined by assuming a density of 0.91 ml/g for myocardium (hence, $V_1 + V_2 = 0.91$ ml/g) [24]. For a given probe, active and fully or partially inactive forms are assumed to have the same uptake kinetics in normal rat myocardium [18].

For MI, the model allows for diffusion of the probe between vascular and tissue regions, and sequestration of probe within the tissue region via specific binding. For ^{99m}Tc-D and ^{99m}Tc-D^{I}, the remaining model parameters (K_{PS}, K_SB_{0A}, and K_SB_{0I}) were estimated by fitting the solution of the governing differential equations (expressed as $C_{total}(t)$) simultaneously to the uptake data of the active and partially inactive form in MI (Figure 3.2A). For ^{99m}Tc-C2A-GST and ^{99m}Tc-C2A-GST-NHS (which is fully inactive) in MI, the value of K_{PS} were first estimated by fitting the model solution with K_SB_{0I} set to zero to the uptake data of the inactive form in normal myocardium (Figure 3.2B). Figure 4.5 shows model fit to ^{99m}Tc-C2A-GST uptake data in normal myocardium. Knowing V_1, V_2, and K_{PS}, the remaining model parameter (K_SB_{0A}) was estimated by fitting the solution of the governing differential equations to the uptake data of the active form in MI (Figure 3.2B).

4.2.2 Model 1 Results:

Figures 4.4 and 4.5 show Model 1 fits to the uptake data of active and partially inactive ^{99m}Tc-D, and active and inactive forms of ^{99m}Tc-C2A-GST in normal and infarct myocardium. Figures 4.4 and 4.5 show a systematic difference between the MI uptake data for both probes and corresponding Model 1 fits. Tables 4.1 and 4.2 show the estimated values for the Model 1 parameters for each of the two probes. Standard errors for estimates of model parameters were determined from sum of squares difference (SSD) between data and model fit and the Jacobian (sensitivity) matrix as previously described [25].

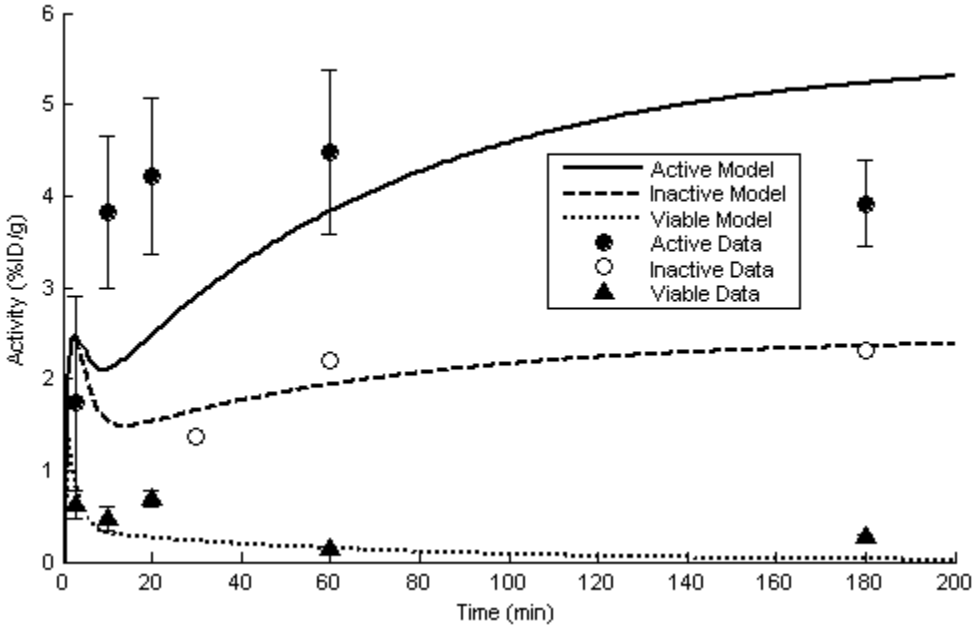


Figure 4.4 Radioactivity (in %ID/g) in MI and normal myocardium for ^{99m}Tc-D at 3 (mean ± S.D.; n = 4), 10 (mean ± S.D.; n = 3), 20 (mean ± S.D.; n = 2), 60 (mean ± S.D.; n = 3), and 180 min (mean ± S.D.; n = 2) after injection. Radioactivity (in %ID/g) in MI for ^{99m}Tc-D^{I} at 30, 60, and 180 min after injection (n = 1). Solid and dashed lines are Model 1 fits to MI uptake data of active and partially inactive probes. Dotted line is Model 1 fit to probe uptake data in normal myocardium.

Table 4.1: Estimated values of Model 1 parameters for ^{99m}Tc-D and measures of precision of these estimates

Parameters	Estimated Value	S.E.
V_1 (ml/g)	0.31	0.08
V_2 (ml/g)	0.61	
K_{PS} (ml/min/g)	0.30	
$(K_S B_0)_A$ (ml/min/g)	0.07	
$(K_S B_0)_I$ (ml/min/g)	0.02	

S.E. = standard error
V_1 = volume of the vascular space
V_2 = volume of the tissue space
K_{PS} = rate of probe diffusion between vascular and tissue spaces
$K_S B_{0A}$ = rate of active probe sequestration within tissue space
$K_S B_{0I}$ = rate of partially inactive probe sequestration within tissue space

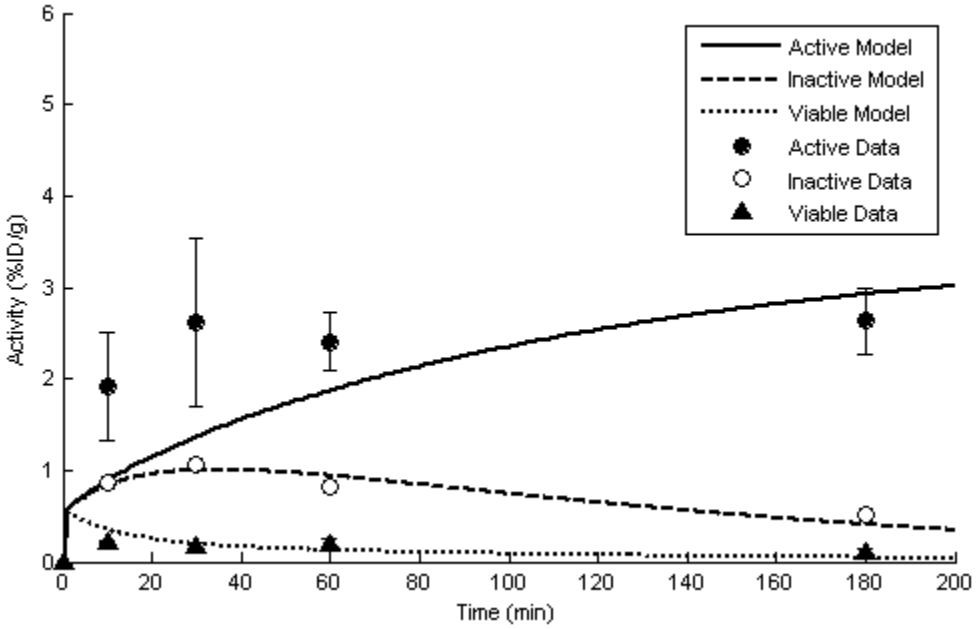


Figure 4.5 Radioactivity (in %ID/g) in MI for ^{99m}Tc-C2A-GST at 10, 20, 60, and 180 min after injection (mean ± S.D.; n = 4). Radioactivity (in %ID/g) in MI for ^{99m}Tc-C2A-GST-NHS at 10, 20, 60, and 180 min after injection (mean ± S.D.; n = 4). Solid and dashed lines are Model 1 fits to MI uptake data of active and fully inactive probes. Dotted line is Model 1 fit to probe uptake data in normal myocardium.

Table 4.2 Estimated values of Model 1 parameters for ^{99m}Tc-C2A-GST and measures of precision of these estimates

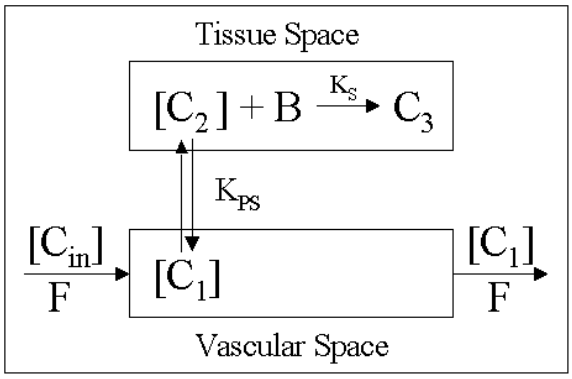
Parameters	Estimated Value	S.E.
V_1 (ml/g)	0.14	<0.01
V_2 (ml/g)	0.78	
K_{PS} (ml/min/g)	0.02	<0.01
$K_S B0_A$ (ml/min/g)	5.95	

S.E. = standard error
V_1 = volume of the vascular space
V_2 = volume of the tissue space
K_{PS} = rate of probe diffusion between vascular and tissue spaces
$K_S B_{0A}$ = rate of active probe sequestration within tissue space

For ^{99m}Tc-D, Model 1 provides a poor fit to the active and partially inactive probe uptake data in MI. For ^{99m}Tc-C2A-GST, the model fits well the fully inactive probe uptake data in MI, but provides a poor fit to the active probe uptake data in MI. Hence, a different model was considered for the uptake data of both probes in MI.

4.3 Model 2:

Model 2 (Figure 4.6) relaxes the assumption that the number of available binding sites for the probe is large as compared to the number of probe molecules available for binding (Model 1). Thus, Model 2 is the same as Model 1 except that it assumes that probe binding in MI to be saturable.



$[C_{in}]$ = input probe concentration
$[C_1]$ = unbound probe concentration in vascular space
$[C_2]$ = unbound probe concentration in tissue space
C_3 = amount of probe bound in the tissue space
F = Flow (1 ml/min/g)

Figure 4.6 Schematic diagram of Model 2 for the uptake of probe in normal and infarct myocardium. K_S is the binding rate constant; K_{PS} is the rate of probe diffusion between the tissue and vascular spaces; B is the total number of binding sites in the tissue space at a given time (t) following the injection of the probe at t = 0.

The governing differential equations for Model 2 are:

$$\frac{d[C_1]}{dt} = \frac{F}{V_1} * ([C_{in}] - [C_1]) + \frac{K_{PS}}{V_1} * ([C_2] - [C_1]) \quad (4.7)$$

$$\frac{d[C_2]}{dt} = \frac{K_{PS}}{V_2} * ([C_1] - [C_2]) - \frac{K_S}{V_2}[C_2]\,(B_0 * m - C_3) \quad (4.8)$$

$$\frac{dC_3}{dt} = K_S * m * [C_2]\,(B_0 * m - C_3) \quad (4.9)$$

As for Model 1, $[C_1]$, $[C_2]$, and C_3 represent the probe concentrations in the vascular and tissue spaces and the amount of bound probe in tissue space, respectively; B_0 is the total number of available tissue binding sites per gram of tissue for the active and partially inactive forms, where $B = B_0*m - C_3$ is the number of available binding sites at time t. K_S is the binding rate constant, which in general may be different for active (K_{SA}) and partially inactive (K_{SI}) forms. Thus, for Model 2, the number of unknown kinetic model parameters is six, namely V_1, V_2, K_{PS}, B_0, K_{SA}, and K_{SI}. For ^{99m}Tc-C2A-GST, K_{SI} is zero since the inactive form does not bind within the tissue space,

As for Model 1, the solution of the model differential equations is expressed as $C_{total}(t)$ which represents the amount of radioactivity in vascular and tissue spaces (in %ID/g).

$$C_{total}(t) = \frac{(V_1 *m * [C_1] + V_2 *m * [C_2] + C_3)(MW * 10^{-9})}{ID * m} * 100 \qquad (4.10)$$

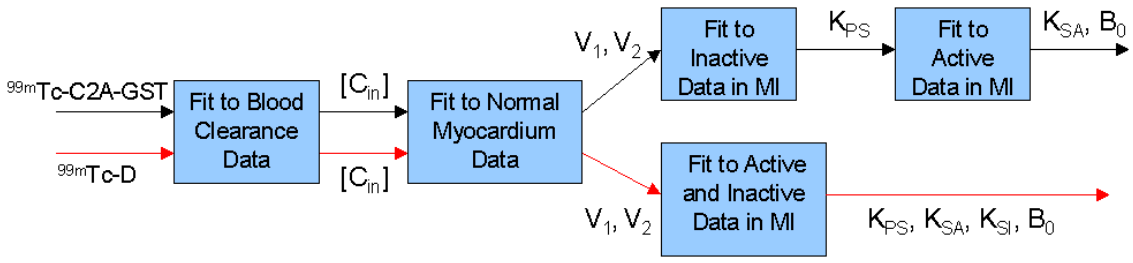


Figure 4.7 Flowchart of the parameter estimation protocol for Model 2. V_1 and V_2 are the volumes of the vascular and tissue spaces, respectively; K_{PS} is the rate of diffusion between the tissue and vascular spaces; K_{SA} and K_{SI} are the binding rate constants for active and partially inactive probe, respectively; B_0 is the total number of available specific binding sites in the tissue space.

4.3.1 Estimation of Model 2 Parameters:

A flowchart of the parameter estimation protocol for Model 2 is shown in Figure 4.7. For both probes, $C_{in}(t)$ and the values of V_1 and V_2 were the same as those estimated

for Model 1. For ^{99m}Tc-D and ^{99m}Tc-D^{I}, the remaining model parameters, namely K_{PS}, B_0, K_{SA}, and K_{SI} were estimated by fitting the model solution (expressed as $C_{total}(t)$) simultaneously to the active and partially inactive form in MI (Figure 3.2A). For ^{99m}Tc-C2A-GST and ^{99m}Tc-C2A-GST-NHS, V_1, V_2 and K_{PS} were estimated as described for Model 1. Knowing the values of V_1, V_2, and K_{PS} (Table 4.2), the values of the remaining model parameters (K_{SA} and B_0) were estimated by fitting the solution of the governing differential equations to the uptake data of the active form in MI (Figure 3.2 B).

4.3.2 Model 2 Results:

Figures 4.8 and 4.9 show Model 2 fits to the uptake data of active and partially inactive ^{99m}Tc-D, and active and inactive forms of ^{99m}Tc-C2A-GST in normal and infarct myocardium. Tables 4.3 and 4.4 show the estimated values for the Model 2 parameters for each of the two probes. Standard errors are derived from the sum of squared difference (SSD) and the Jacobian matrix obtained from the parameter estimation. Correlation matrices are derived from the Jacobian matrix obtained from the parameter estimation. These correlation matrices provide a measure of the degree of relationship between a pair of parameters.

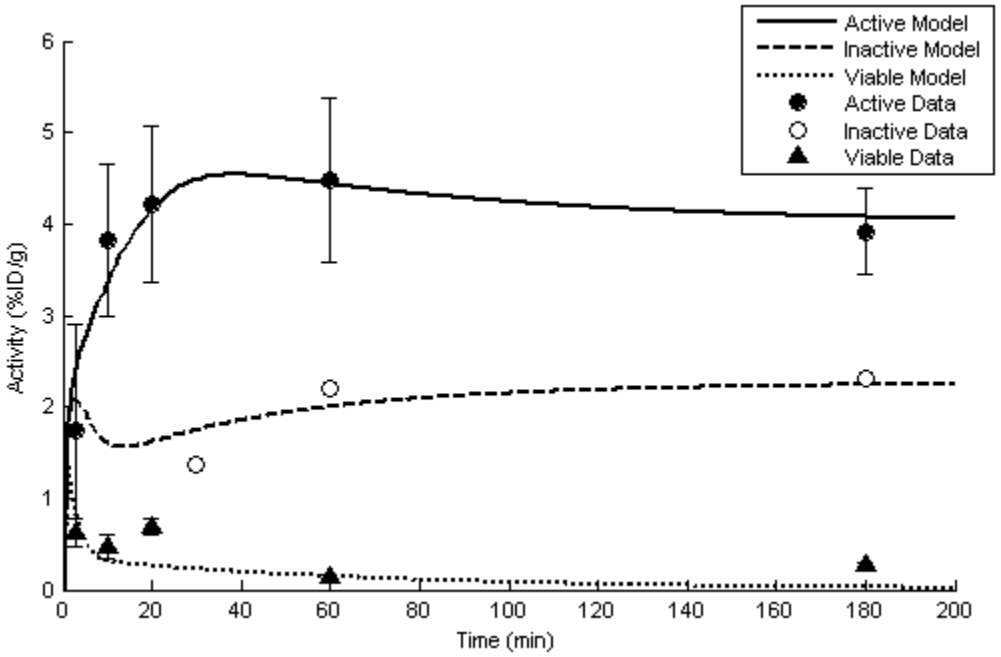


Figure 4.8 Radioactivity (in %ID/g) in MI for ^{99m}Tc-D at 3 (mean ± S.D.; n = 4), 10 (mean ± S.D.; n = 3), 20 (mean ± S.D.; n = 2), 60 (mean ± S.D.; n = 3), and 180 min (mean ± S.D.; n = 2) after injection. Radioactivity (in %ID/g) in MI for ^{99m}Tc-D^{I} at 30, 60, and 180 min after injection (n = 1). Solid and dashed lines are Model 2 fits to MI uptake data of active and partially inactive probes. Dotted line is Model fit to probe uptake data in normal myocardium.

Table 4.3 Estimated values of Model 2 parameters for ^{99m}Tc-D and measures of precision of these estimates

Parameters	Estimated Value	S.E.		Correlation Matrix			
K_{PS} (ml/min/g)	0.25	0.13		1.00	-0.93	-0.40	0.15
K_{SA} (ml/nmol/min/g)	>50				1.00	0.43	-0.28
K_{SI} (ml/nmol/min/g)	2.94	0.74				1.00	-0.43
B_0 (nmol/g)	0.13	0.01					1.00

The ijth entry of the correlation matrix is the correlation coefficient between the ith and jth parameter.
S.E. = standard error
K_{PS} = rate of probe diffusion between vascular and tissue spaces
K_{SA} = active probe binding rate constant
K_{SI} = partially inactive probe binding rate constant
B_0 = total number of available binding sites

Unlike Model 1, Model 2 appears to provide a reasonably good fit to the ^{99m}Tc-D uptake data in MI. For ^{99m}Tc-D, the estimated value of K_{SA} was 50 ml/nmol/min/g, which was set as an upper bound on the value of this parameter. The correlation matrix (Table

4.3), determined as previously described [25], reveals a relatively high correlation (0.93) between K_{PS} and K_{SA} which suggests that the effect of the rate of diffusion into the tissue and the binding rate of the probe cannot be easily distinguished.

Fisher's F-test was used to confirm that the Model 2 provides a statistically better fit than Model 1 [25]. Below is the equation used to calculate F for two models with different numbers of parameters:

$$F = \frac{\left(SSD_1 - SSD_2\right) / \left(P_2 - P_1\right)}{SSD_2 / \left(N - P_2\right)} \tag{4.11}$$

where SSD_1 and SSD_2 are the sum of squared differences of the best fit for Model 1 and Model 2, respectively; P_1 and P_2 are the numbers of parameters for Model 1 and Model 2, respectively ($P_2 > P_1$); and N is the number of data points to which the solutions of Models 1 and 2 were each fit to (Figures 4.4 and 4.8). The calculated F value was compared to a critical value from an F-distribution table to determine whether to reject or accept the null hypothesis that Model 2 does not provide a statistically better fit than Model 1. The results of this F-test demonstrated that the Model 2 fit is statistically superior to Model 1 fit.

In order to evaluate the sensitivity of the above model fit and estimated values of model parameters to the upper bound of 50 ml/nmol/min/g set for K_{SA}, Model 2 was refitted to the ^{99m}Tc-D uptake data with the upper bound set at 40, 30, 20, 15 ml/min/g. To compare Model fits at different upper bounds for the value of K_{SA}, the following version of the F-test equation was used:

$$F = \frac{SSD_{2A}}{SSD_{2B}} \tag{4.12}$$

where SSD_{2B} is the sum of squared difference of Model 2 fit with the upper at bound set at 50 ml/nmol/min/g and SSD_{2A} is the sum of squared difference of Model 2 fit with the upper bound at set at < 50 ml/nmol/min/g. The results of this F-test demonstrated that a value of < 15 ml/nmol/min/g for K_{SA} results in a Model 2 fit that is significantly worse than the fit with the upper bound set at 50 ml/nmol/min/g.

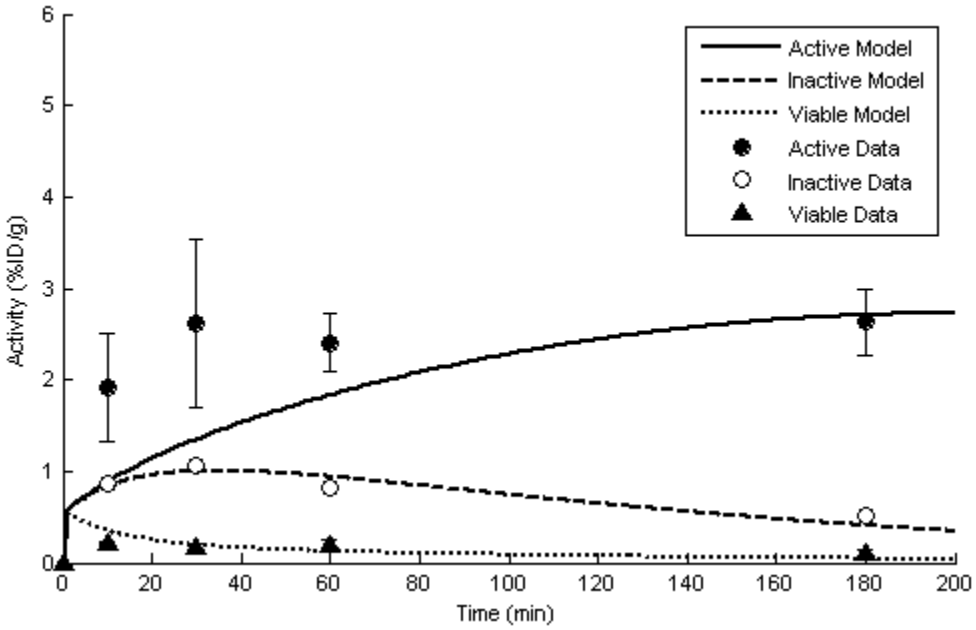


Figure 4.9 Radioactivity (in %ID/g) in MI for ^{99m}Tc-C2A-GST at 10, 20, 60, and 180 min after injection (mean ± S.D.; n = 4). Radioactivity (in %ID/g) in MI for ^{99m}Tc-C2A-GST-NHS at 10, 20, 60, and 180 min after injection (mean ± S.D.; n = 4). Solid and dashed lines are Model 2 fits to MI uptake data of active and inactive probes. Dotted line is Model fit to probe uptake data in normal myocardium.

Table 4.4 Estimated values of Model 2 parameters for ^{99m}Tc-C2A-GST

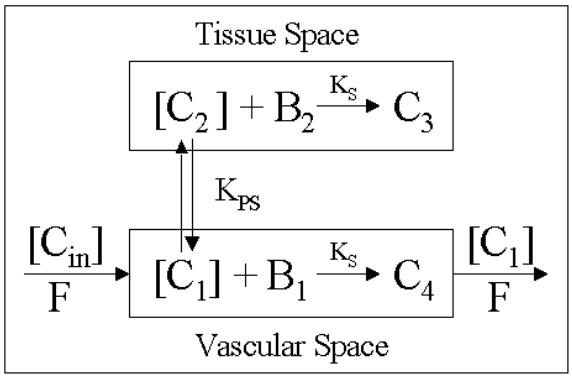
Parameters	Estimated Value
K_{SA} (ml/nmol/min/g)	>50
B_0 (nmol/g)	0.05

K_{SA} = active probe binding rate constant
B_0 = total number of available binding sites

Like Model 1, Model 2 does not provide good fit to the ^{99m}Tc-C2A-GST uptake data in MI for active probe. Both models failed to fit the early phase of the uptake data in MI (Figure 3.2B).

4.4 Model 3:

Model 3 was developed by Audi et al and used to evaluate ^{99m}Tc-C2A-GST uptake kinetics in normal and infarct myocardium [18]. This model allows for saturable binding of active probe in both vascular and tissue spaces. For each form of the probe (active and fully or partially inactive), the binding rate constant of the probe with binding sites in the vascular space was assumed to be the same as that with binding sites in the tissue space [18].



$[C_{in}]$ = input probe concentration
$[C_1]$ = unbound probe concentration in vascular space
$[C_2]$ = unbound probe concentration in tissue space
C_3 = amount of probe bound in the tissue space
C_4 = amount of probe bound in the vascular space
F = Flow (1 ml/min/g)

Figure 4.10 Schematic diagram of Model 3 for the uptake of probe in normal and infarct myocardium. K_S is the binding rate constant; K_{PS} is the rate of probe diffusion between the tissue and vascular spaces; B_1 and B_2 are the numbers of available binding sites in the vascular and tissue

The governing differential equations for Model 3 are:

$$\frac{d[C_1]}{dt} = \frac{F}{V_1} * ([C_{in}] - [C_1]) + \frac{K_{PS}}{V_1} * ([C_2] - [C_1]) - \frac{K_S}{V_1}[C_1]\,(B_{01} * m - C_4) \qquad (4.13)$$

$$\frac{d[C_2]}{dt} = \frac{K_S}{V_2} * ([C_1] - [C_2]) - \frac{K_S}{V_2}[C_2]\,(B_{02} * m - C_3) \qquad (4.14)$$

$$\frac{dC_3}{dt} = K_S * m * [C_2]\,(B_{02} * m - C_3) \qquad (4.15)$$

$$\frac{dC_4}{dt} = K_S * m * [C_1]\,(B_{01} * m - C_4) \qquad (4.16)$$

As in Models 1 and 2, $[C_1]$, $[C_2]$, and C_3 represent the probe concentrations in the vascular and tissue spaces and the amount of bound probe in tissue space, respectively. Additionally, C_4 represents the amounts of bound probe in the vascular space. B_{01} and B_{02} represent the total amounts of available binding sites in the vascular and tissue spaces, respectively. $B_1 = B_{01}*m - C_4$ and $B_2 = B_{02}*m - C_3$ represent the free binding sites at time t in the vascular and tissue spaces, respectively. For Model 3, the number of kinetic parameters is seven, namely V_1, V_2, K_{PS}, K_{SA}, K_{SI}, B_{01}, and B_{02}. As for Models 1 and 2, the solution of the model differential equations is expressed as $C_{total}(t)$ which represents the amount of radioactivity in vascular and tissue spaces (in %ID/g).

$$C_{total}(t) = \frac{(V_1 * m * [C_1] + V_2 * m * [C_2] + C_3 + C_4)(MW * 10^{-9})}{ID * m} * 100 \qquad (4.17)$$

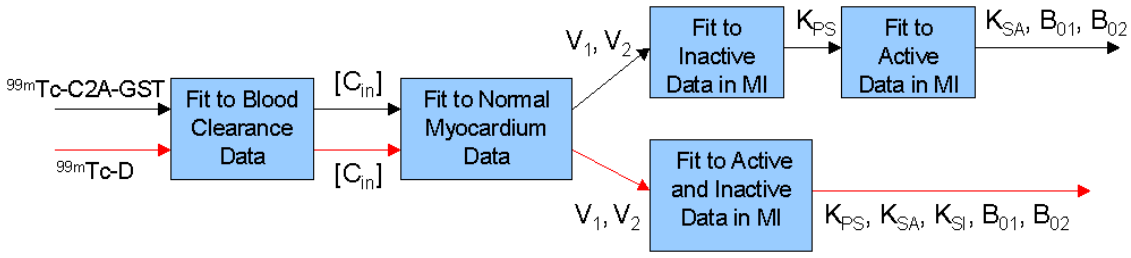


Figure 4.11 Parameter estimation protocol for Model 3. V_1 and V_2 are the volumes of the vascular and tissue spaces, respectively; K_{PS} is the rate of diffusion between the tissue and vascular spaces; K_{SA} and K_{SI} are the binding rate constants for active and partially inactive probe, respectively; B_{01} and B_{02} are the total number of specific binding sites in the vascular and tissue spaces, respectively.

4.4.1 Estimation of Model 3 Parameters:

A flowchart of the parameter estimation protocol for Model 3 is shown in Figure 4.11. For both probes, $C_{in}(t)$ and the values of V_1 and V_2 were the same as those estimated for Model 1. For ^{99m}Tc-D and ^{99m}Tc-D^{I}, the remaining model parameters, namely K_{PS}, B_{01}, B_{02}, K_{SA}, and K_{SI} were estimated by fitting the model solution (expressed as $C_{total}(t)$) simultaneously to the active and partially inactive form in MI (Figure 3.2A). For ^{99m}Tc-C2A-GST and ^{99m}Tc-C2A-GST-NHS, V_1, V_2 and K_{PS} were estimated as described for Model 1. Knowing the values of V_1, V_2, and K_{PS} (Table 4.2), the values of the remaining model parameters (K_{SA}, B_{01}, and B_{02}) were estimated by fitting the solution of the governing differential equations to the uptake data of the active form in MI (Figure 3.2 B).

4.4.2 Model 3 Results

Figures 4.12 and 4.13 show Model 3 fits to the uptake data of active and partially inactive ^{99m}Tc-D, and active and inactive forms of ^{99m}Tc-C2A-GST in normal and infarct myocardium. Tables 4.5, 4.6, and 4.7 show the estimated values for the Model 3 parameters for each of the two probes.

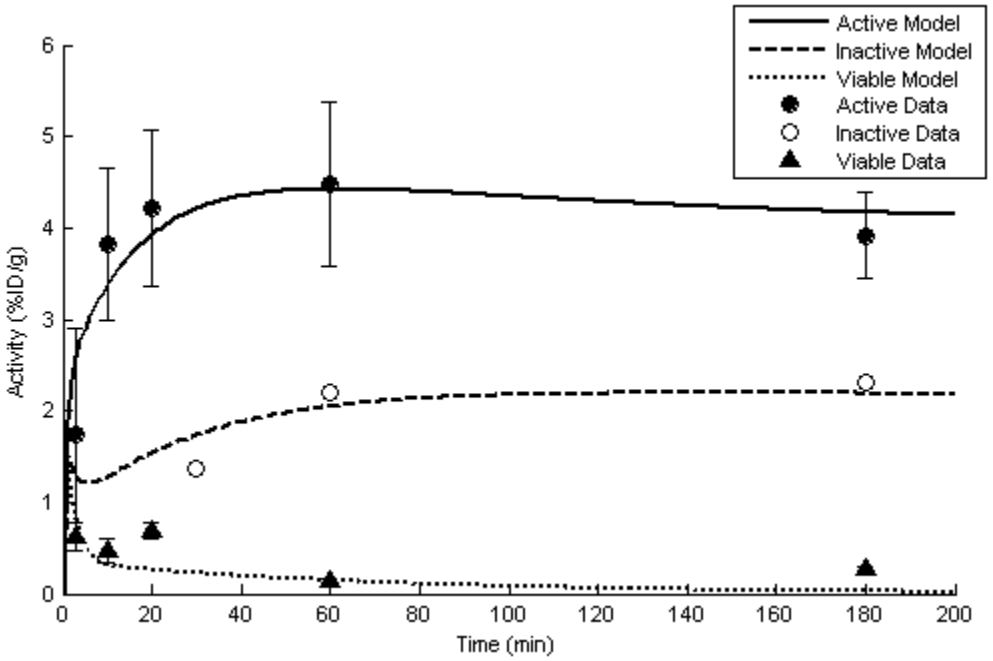


Figure 4.12 Radioactivity (in %ID/g) in MI for ^{99m}Tc-D at 3 (mean ± S.D.; n = 4), 10 (mean ± S.D.; n = 3), 20 (mean ± S.D.; n = 2), 60 (mean ± S.D.; n = 3), and 180 min (mean ± S.D.; n = 2) after injection. Radioactivity (in %ID/g) in MI for ^{99m}Tc-D^{I} at 30, 60, and 180 min after injection (n = 1). Solid and dashed lines are Model 3 fits to MI uptake data of active and partially inactive probes. Dotted line is Model fit to probe uptake data in normal myocardium.

Table 4.5 Estimated values of Model 3 parameters for ^{99m}Tc-D and measures of precision of these estimates

Parameters	Estimated Value	S.E.		Correlation Matrix				
K_{PS} (ml/min/g)	0.02	0.07		1.00	0.28	-0.03	-0.77	0.81
K_{SA} (ml/nmol/min/g)	15.53	11.42			1.00	0.54	-0.79	0.72
K_{SI} (ml/nmol/min/g)	2.40	0.77				1.00	-0.35	0.26
B_{01} (nmol/g)	0.13	0.07					1.00	-0.98
B_{02} (nmol/g)	0.00	0.08						1.00

The ijth entry of the correlation matrix is the correlation coefficient between the ith and jth parameter.
S.E. = standard error
K_{PS} = rate of probe diffusion between vascular and tissue spaces
K_{SA} = active probe binding rate constant
K_{SI} = partially inactive probe binding rate constant
B_{01} = total number of available binding sites in the vascular space
B_{02} = total number of available binding sites in the tissue space

Like Model 2, Model 3 provided a reasonably good fit to the ^{99m}Tc-D uptake data in MI. The results reveal that no binding of probe in the tissue space is necessary to describe probe uptake in MI. Model 3 was refit to the ^{99m}Tc-D uptake data with B_{02} set to zero. As expected, this refit of Model 3 gave the same fit and estimated values for model parameter as those with B_{02} free (Table 4.6).

Table 4.6 Estimated values of Model 3 parameters (with B_{02} set to zero) for ^{99m}Tc-D and measures of precision of these estimates

Parameters	Estimated Value	S.E.		Correlation Matrix			
K_{PS} (ml/min/g)	0.02	0.04		1.00	0.28	-0.03	-0.77
K_{SA} (ml/nmol/min/g)	15.16	7.46			1.00	0.54	-0.79
K_{SI} (ml/nmol/min/g)	2.38	0.68				1.00	-0.35
B_{01} (nmol/g)	0.13	0.01					1.00

The ijth entry of the correlation matrix is the correlation coefficient between the ith and jth parameter.
S.E. = standard error
K_{PS} = rate of probe diffusion between vascular and tissue spaces
K_{SA} = active probe binding rate constant
K_{SI} = partially inactive probe binding rate constant
B_{01} = total number of available vascular space binding sites

The F-test was again performed to compare Model 2 and Model 3 which have the same number of parameters since B_{02} is set to zero. Essentially, Model 2 and Model 3 are the same model under this assumption, with the exception that Model 2 allows for binding in the tissue space only and Model 3 allows for binding in the vascular space only. For two models with the same number of parameters, the following F equation is used:

$$F = \frac{SSD_2}{SSD_3} \tag{4.18}$$

where SSD_2 and SSD_3 are the sum of squared differences of the best fit for Model 1 and Model 2 , respectively. Results of this F-test revealed that Model 3 fit to ^{99m}Tc-D uptake data in MI is not statistically superior to Model 2 fit.

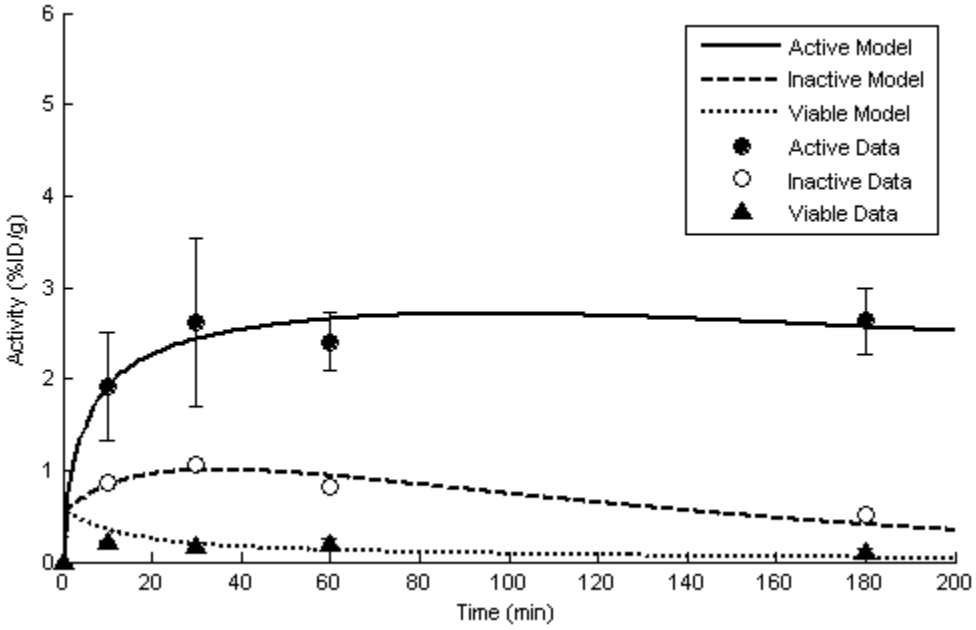


Figure 4.13 Radioactivity (in %ID/g) in MI for ^{99m}Tc-C2A-GST at 10, 20, 60, and 180 min after injection (mean ± S.D.; n = 4). Radioactivity (in %ID/g) in MI for ^{99m}Tc-C2A-GST-NHS at 10, 20, 60, and 180 min after injection (mean ± S.D.; n = 4). Solid and dashed lines are Model 3 fits to MI uptake data of active and inactive probes. Dotted line is Model fit to probe uptake data in normal myocardium

Table 4.7 Estimated values of Model 3 parameters for ^{99m}Tc-C2A-GST and measures of precision of these estimates

Parameters	Estimated Value	S.E.		Correlation Matrix		
K_{SA} (ml/nmol/min/g)	6.17	6.70		1.00	-0.92	0.78
B_{01} (nmol/g)	0.03	0.01			1.00	-0.90
B_{02} (nmol/g)	0.02	0.01				1.00

The ijth entry of the correlation matrix is the correlation coefficient between the ith and jth parameter.
S.E. = standard error
K_{SA} = active probe binding rate constant
B_{01} = total number of available vascular space binding sites
B_{02} = total number of available tissue space binding sites

Model 3 provides a better fit to ^{99m}Tc-C2A-GST uptake data as compared to both Model 1 and Model 2. As previously reported by Audi et al, a model which accounts for non-linear binding in the tissue and vascular spaces is sufficient to describe the uptake of ^{99m}Tc-C2A-GST in MI. The F-test was again used to compare the ability of Model 3 and Model 2 to fit to the ^{99m}Tc-C2A-GST uptake data. The F-test showed that the Model 3 fit was statistically superior to the fits of Models 1 and 2. This result suggests that both tissue space and vascular space binding sites are needed to account for the uptake of ^{99m}Tc-C2A-GST in MI.

4.5 Comparison of Estimated Values of Model Parameters for ^{99m}Tc-D and ^{99m}Tc-C2A-GST:

A summary of the kinetic model parameters for Models 2 and 3 is shown in Table 4.8.

Table 4.8 Summary of parameters from Model 2 and Model 3

Parameters	Model 2	Model 3
Volumes	V_1, V_2	V_1, V_2
Probe Tissue Permeability	K_{PS}	K_{PS}
Binding Kinetics	K_{SA}, K_{SI}, B_0	K_{SA}, K_{SI}, B_{01}, B_{02}

V_1 = volume of the vascular space
V_2 = volume of the tissue space
K_{PS} = rate of probe diffusion between vascular and tissue spaces
K_{SA} = active probe binding rate constant
K_{SI} = partially inactive probe binding rate constant
B_0 = total number of available binding sites (Model 2)
B_{01} = total number of available binding sites in the vascular space (Model 3)
B_{02} = total number of available binding sites in the tissue space (Model 3)

Again, one of the questions addressed in this study is how much of the difference in uptake of ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST in MI is due differences in the binding kinetics (i.e. binding rate constant and/or number of available binding sites) and how due to the fact that ^{99m}Tc-D has a molecular weight (3 kDa) that is significantly

smaller than that of ^{99m}Tc-C2A-GST (37 kDa) which would be expected to correspond to a higher tissue permeability of ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST [14]. The contribution of membrane permeability to the difference in uptake of ^{99m}Tc-C2A-GST vs. ^{99m}Tc-D in MI was determined by comparing the estimated values of K_{PS} for each probe. The contribution of the binding kinetics to the difference in uptake of ^{99m}Tc-C2A-GST and ^{99m}Tc-D in MI was determined by comparing the binding constant (K_{SA}) and the total number of binding sites (B_0 for Model 2, $B_{01} + B_{02}$ for Model 3) for each probe.

Table 4.9 Comparison of uptake parameters

Uptake Factors	Parameters	^{99m}Tc-D Model 2	^{99m}Tc-D Model 3	^{99m}Tc-C2A-GST Model 3
Permeability	K_{PS} (ml/min/g)	0.25	0.02	0.02
Binding Kinetics	K_{SA} (ml/nmol/min/g)	>15.00	15.16	6.17
	B_0 or ($B_{01} + B_{02}$) (nmol/g)	0.13	0.13	0.05

K_{PS} = rate of probe diffusion between vascular and tissue spaces
K_{SA} = active probe binding rate constant
B_0 = total number of available binding sites (Model 2)
B_{01} = total number of available vascular space binding sites (Model 3)
B_{02} = total number of available tissue space binding sites (Model 3)

Table 4.9 compares the estimated values of Model 2 and 3 parameters descriptive of probe permeability and binding kinetics for ^{99m}Tc-D and ^{99m}Tc-C2A-GST. The estimated value of K_{PS} from Model 2 for ^{99m}Tc-D is nearly 13-fold larger than the corresponding value from Model 3 for ^{99m}Tc-C2A-GST. However, the K_{PS} value from Model 3 for ^{99m}Tc-D was nearly the same as that of Model 3 for ^{99m}Tc-C2A-GST. When comparing binding kinetics, Model 2 fit for ^{99m}Tc-D estimated a very high value for K_{SA} that was only limited by the upper bound set on the value of this parameter, whereas

Model 3 fit for ^{99m}Tc-D estimated a value for K_{SA} that is nearly 3-fold larger than that estimated by fitting Model 3 to the ^{99m}Tc-C2A-GST. The total number of available binding sites (B_0 for Model 2 or (B_{01} + B_{02}) for Model 3) estimated from Model 2 and Model 3 for ^{99m}Tc-D were nearly the same and were about 3-fold larger compared to the total number of available binding sites estimated from Model 3 for ^{99m}Tc-C2A-GST.

CHAPTER V
DISCUSSION AND CONCLUSIONS

The objective of this thesis was to establish a quantitative basis for interpretation of ^{99m}Tc-D *in vivo* uptake in normal and infarct myocardial tissue, and for comparison of the myocardial uptake kinetics of ^{99m}Tc-D with other molecular probes of myocardial infarction (MI). The specific aims were 1) to develop a compartmental model for the uptake kinetics of ^{99m}Tc-D in normal and infarct myocardium, and 2) utilize this compartmental model to better understand the underlying mechanisms for the difference in the uptake kinetics of ^{99m}Tc-D and ^{99m}Tc-C2A-GST in rat MI.

Three compartmental models were evaluated for their ability to describe the uptake of ^{99m}Tc-D and ^{99m}Tc-C2A-GST and their fully or partially inactive forms in normal and infarct myocardium. For ^{99m}Tc-D and its partially inactive form (^{99m}Tc-D^{I}), Model 2 and Model 3 were both shown to provide reasonable fits to the uptake data in MI. The key difference between the two models is that Model 2 allows for specific binding of the probe only in the tissue space, whereas Model 3 allows for specific binding of probe in both vascular and tissue spaces. Model 3 results show that specific binding of probe in the vascular space alone is sufficient to explain the uptake kinetics of ^{99m}Tc-D and ^{99m}Tc-D^{I} in normal and infarct rat myocardium. By the F-test, both models were shown to be equivalent in their ability to fit the uptake data.

Model 3 was shown to provide a statistically superior fit to the uptake data of ^{99m}Tc-C2A-GST in MI as compared to Models 1 and 2. Model 3 fit demonstrates the need for nonlinear specific binding of ^{99m}Tc-C2A-GST in both the tissue and vascular spaces. This result is consistent with previous work by Audi et al since Model 3 was

previously used to fit to the same uptake data of ^{99m}Tc-C2A-GST in normal and infarct rat myocardium [18].

The total number of specific binding sites is a measure of necrosis and/or apoptosis in MI. The number of available specific binding sites for ^{99m}Tc-D and ^{99m}Tc-C2A-GST in MI estimated using Model 2 (0.13 nmol/g) and Model 3 (0.13 nmol/g) are ~3-fold larger than the number of available specific binding sites for ^{99m}Tc-C2A-GST in MI (0.05 nmol/g) estimated using Model 3. This result is consistent with the fact that PE (the binding target for ^{99m}Tc-D) accounts for ~20% of all phospholipids in mammalian cellular membranes as compared to ~7% for PS and PI (the binding targets for ^{99m}Tc-C2A-GST) combined. Thus, despite differences in the properties (MWs, binding rate constants and available binding sites) of ^{99m}Tc-D and ^{99m}Tc- C2A-GST, they appear to detect a consistent level of cell death in the same model of MI.

The ~3-fold difference in the number of available binding sites for ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST resulted in only ~48% increase in the uptake of ^{99m}Tc-D in MI compared to ^{99m}Tc-C2A-GST at steady-state (Figure 3.2). This result is in part due to the faster blood clearance rate of ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST (Figure 3.1). This stresses the importance of the modeling approach for the quantitative interpretation of ^{99m}Tc-D uptake data since the ~3-fold difference in number of available binding sites cannot be predicted directly from the uptake kinetic data (Figure 3.2) without accounting for the contributions of the other factors that determine the uptake kinetics of ^{99m}Tc-D in MI, including blood clearance rate.

When scaled by the number of cells per gram of tissue (~100 x 10^6 cells/g), the estimate of 0.13 nmol/g suggests that there are ~800,000 PE molecules per cell [26, 27].

It has been previously reported that there are ~10^6 PS molecules per cell, which would suggest that there are ~9 x 10^6 total PE molecules per cell since PE accounts for ~20% of all phospholipids [28]. The difference between the model estimate of available PE binding sites per cell and the reported number suggests that some of the PE binding sites may not be accessible to ^{99m}Tc-D from the vascular space.

The Model 2 estimate of the binding rate constant (K_{SA}) for ^{99m}Tc-D (>15 ml/nmol/min/g) is larger than that estimated for ^{99m}Tc-C2A-GST using Model 3 (6.2 ml/nmol/min/g). A large binding rate constant is needed in order to sequester as much of the probe as possible before it is cleared from the blood. Like Model 2, Model 3 predicts a larger binding rate constant, K_{SA} for ^{99m}Tc-D (K_{SA} = 15.2 ml/nmol/min/g) compared to ^{99m}Tc-C2A-GST (K_{SA} = 6.2 ml/nmol/min/g). Thus, Models 2 and 3 suggest that a larger binding rate constant for ^{99m}Tc-D is important for the greater uptake of ^{99m}Tc-D in MI as compared to ^{99m}Tc-C2A-GST.

Model 2 suggests that the difference in the uptake of ^{99m}Tc-D in MI as compared to ^{99m}Tc-C2A-GST is due not only to a ~3-fold change in the number of available binding sites and a greater binding rate constant, but also due to a greater tissue permeability for ^{99m}Tc-D. Tables 4.2 and 4.3 show that the estimate of tissue permeability (K_{PS}) for ^{99m}Tc-D (K_{PS} = 0.25 ml/min/g) is 13-fold larger than that for ^{99m}Tc-C2A-GST (K_{PS} = 0.02 ml/min/g). This difference suggests that the low molecular weight of ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST greatly facilitates the ability of the probe to access the binding sites present in the tissue space and is key to the high early uptake of the probe in MI. The low molecular weight of ^{99m}Tc-D also accounts for its faster blood clearance which causes the probe to enter the system faster but also leave the system faster, leaving

less time for the probe to reach binding sites for sequestration in the tissue space. Sensitivity analysis shows that the early phase of the uptake of ^{99m}Tc-D in MI is most sensitive to K_{PS}, whereas the late phase (plateau) is most sensitive to the number of available binding sites (B_0).

In contrast to Model 2, Model 3 resulted in an estimate of the tissue permeability parameter, K_{PS}, for ^{99m}Tc-D (K_{PS} = 0.02 ml/min/g) that is the same as that estimated for ^{99m}Tc-C2A-GST (K_{PS} = 0.02 ml/min/g). This could be in part due to the lack of sensitivity of the data to this parameter based on the Model 3 assumption that there is saturable binding to specific binding sites in the vascular space. This lack of sensitivity is not surprising since^{99m}Tc-D diffusion into the tissue space is not required for its sequestration in MI since tissue binding was not needed for Model 3 to fit the uptake data. Thus, Model 3 results for ^{99m}Tc-D suggest that ^{99m}Tc-D relatively small molecular weight is not important to the increase in the uptake of ^{99m}Tc-D in MI as compared to ^{99m}Tc-C2A-GST.

The Model 2 prediction of a 13-fold larger K_{PS} for ^{99m}Tc-D as compared to ^{99m}Tc-C2A-GST is much larger than what would be predicted based on the 12-fold difference in molecular weight. For a neutral compound, the diffusion coefficient is inversely proportional to the cubic root of the molecular weight according to the Stokes-Einstein relation [29]. Thus, the 12-fold difference in molecular weight would be expected to result in a K_{PS} for ^{99m}Tc-D that is ~3-fold that for ^{99m}Tc-C2A-GST. To determine the sensitivity of Model 2 fit to ^{99m}Tc-D to the value of K_{PS}, Model 2 was refit to the ^{99m}Tc-D uptake data with K_{PS} set to 3-, 5- and 7-fold the value estimated for ^{99m}Tc-C2A-GST (K_{PS}

= 0.02 ml/min/g). Figures 5.1 and 5.2 show the model fits to ^{99m}Tc-D uptake data in MI with K_{PS} fixed for Models 2 and 3, respectively.

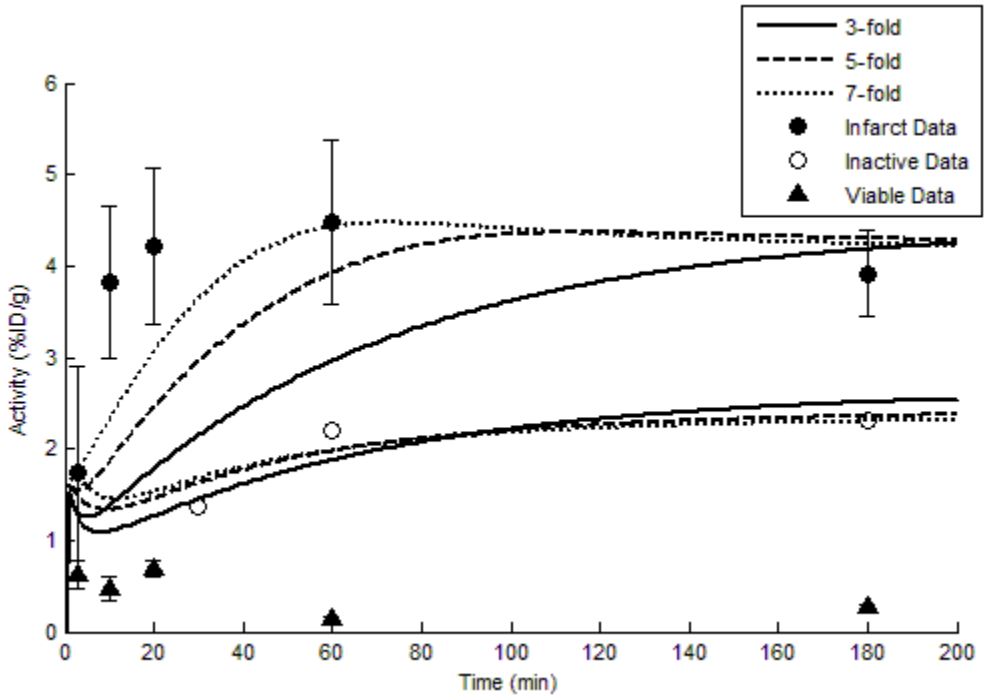


Figure 5.1 Radioactivity (in %ID/g) in MI for ^{99m}Tc-D at 3 (mean ± S.D.; n = 4), 10 (mean ± S.D.; n = 3), 20 (mean ± S.D.; n = 2), 60 (mean ± S.D.; n = 3), and 180 min (mean ± S.D.; n = 2) after injection. Radioactivity (in %ID/g) in MI for ^{99m}Tc-D^{I} at 30, 60, and 180 min after injection (n = 1). Solid, dashed, and dotted lines are Model 2 fits to active and partially inactive probe with K_{PS} at 3-, 5-, and 7-fold the value estimated for ^{99m}Tc-C2A-GST.

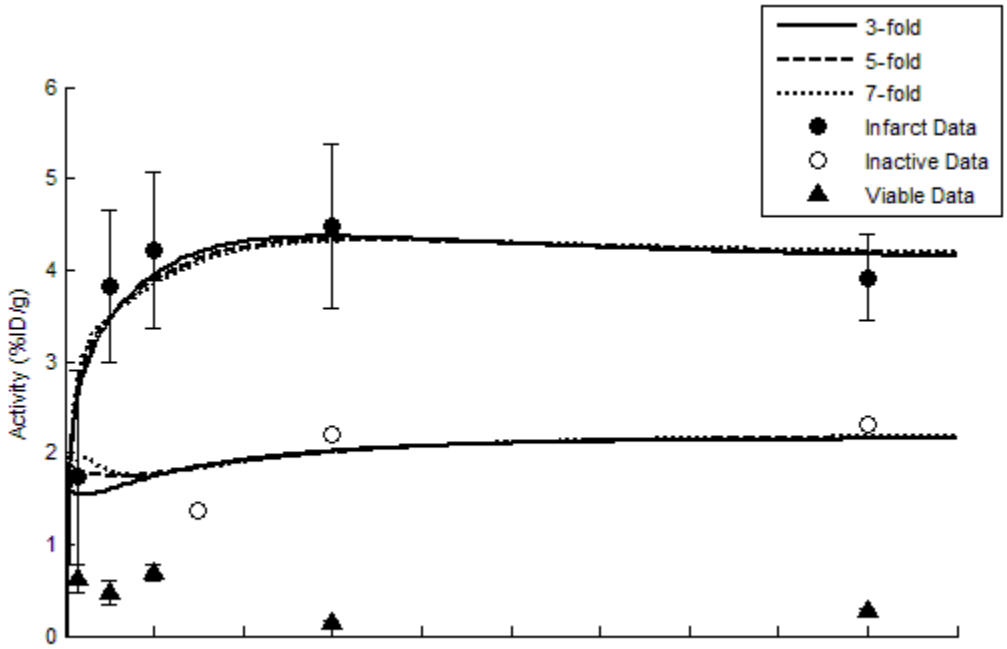


Figure 5.2 Radioactivity (in %ID/g) in MI for ^{99m}Tc-D at 3 (mean ± S.D.; n = 4), 10 (mean ± S.D.; n = 3), 20 (mean ± S.D.; n = 2), 60 (mean ± S.D.; n = 3), and 180 min (mean ± S.D.; n = 2) after injection. Radioactivity (in %ID/g) in MI for ^{99m}Tc-D^{I} at 30, 60, and 180 min after injection (n = 1). Solid, dashed, and dotted lines are Model 3 fit to active and partially inactive probe with K_{PS} at 3-, 5-, and 7-fold the value estimated for ^{99m}Tc-C2A-GST.

Figure 5.1 shows that for all three K_{PS} values, Model 2 fit to the ^{99m}Tc-D data is significantly worse than that obtained with K_{PS} at 13-fold. On the other hand, setting K_{PS} at 3-, 5- and 7-fold the value estimated for ^{99m}Tc-C2A-GST had no significant effect of Model 3 fit to the ^{99m}Tc-D data. Thus, factors other than molecular weight (e.g., charge distribution) could affect the tissue permeation of these probes. For instance, ^{99m}Tc-C2A-GST is known to have a positive net charge which facilitates its binding to negatively charged membrane phospholipids [30, 31]. ^{99m}Tc-D on the other hand is neutral which would facilitate its diffusion between vascular and tissue spaces. This could account for the larger than expected difference in membrane permeability between ^{99m}Tc-D and ^{99m}Tc-C2A-GST predicted by the Stokes-Einstein equation based on the difference in the molecular weights of these probes [32].

One important limiting factor of this study is the lack of a fully inactive form of ^{99m}Tc-D. The development of a fully inactive form would allow for an independent estimate of K_{PS} for ^{99m}Tc-D and potentially break the correlation between K_{PS} and K_S and hence provide a more robust analysis of the contributions of the processes that these parameters represent to the uptake of ^{99m}Tc-D in MI. Moreover, the existence of a fully inactivated form may allow us to distinguish between Models 2 and 3 based on their ability to fit the uptake data of the fully inactivated form.

For both Model 2 and 3 fits to ^{99m}Tc-D, the standard errors (SE) calculated for K_{PS} and K_{SA} were large compared to the estimated values for these parameters. One possible reason could be the small number of data points as compared to number of model parameters. Thus, SE could be decreased by the availability of more uptake data points. The large SE for K_{PS} and K_{SA} can also be attributed to the fact that both parameters affect the early phase of the uptake in MI which exhibits a fast change from zero to the plateau within 20 minutes. An increase in data points in the early phase of the uptake, between 0 and 20 minutes, may allow for better confidence in the estimates of K_{PS} and K_{SA} by providing a more defined representation of the initial rise in uptake of the probe in MI.

Another opportunity for future work would be to explore the uptake of ^{99m}Tc-D in ischemic tissue. The MI tissue is usually surrounded by ischemic tissue often referred to as hibernating myocardium. This peri-infarct ischemic tissue exists in a new functional state that matches the low energy supply with a downregulation in energy utilization. The ischemic tissue is still repairable and serves as a critical reserve of dysfunctional cells that may be potentially rescued. Uptake data from the ischemic tissue has been

shown to have a rapid increase in the early stage with a decrease at 60 minutes before rising again at 180 minutes. The peri-infarct ischemic tissue presents a time-dependent deterioration into infarct myocardium. This change from ischemic to infarct myocardium can be represented in the model by time-dependent diffusion rate and binding parameters.

The long term objective of this thesis is to establish a quantitative basis for interpretation of ^{99m}Tc-D uptake in normal infarct myocardium measured *in vivo* instead of the *in vitro* approach in the present study. One the inputs for the modeling approach developed in this thesis is the mass (m) of the infarct tissue, which cannot be determined in vivo. This mass is needed to determine the fraction of the coronary circulation blood flow (and hence the fraction of the injected dose) that perfuses the infarct region. Otherwise, the model and the parameter estimation steps described for the interpretation of the *in vitro* data could be used for the interpretation of *in vivo* data. One approach for estimating the fraction of the coronary circulation blood flow that perfuses the infarct region would be to inject another SPECT probe for perfusion that would provide information about the fraction of the cardiac output perfusing the infarct region [33]. Additional studies would be needed to evaluate the utility of such approach in this model of MI.

In conclusion, the results of this study provide a mathematical model approach for the quantitative interpretation of the uptake of ^{99m}Tc-D and ^{99m}Tc-C2A-GST in MI and for determining the contributions of various factors to the differences in the uptake of these and future probes in MI. Kinetic analysis suggests that a 3-fold increase in the number of available binding sites and the binding rate constant contributed to the increase in uptake for ^{99m}Tc-D in MI as compared to ^{99m}Tc-C2A-GST. Model 2 also suggests that

an increase in ^{99m}Tc-D tissue permeability, presumably due to its small molecular weight, is also important to the increase in the uptake of ^{99m}Tc-D in MI as compared to ^{99m}Tc-C2A-GST.

BIBLIOGRAPHY

[1] M. N. Levy, A. J. Pappano (2007). Cardiovascular Physiology. Philadelphia, PA, Mosby Inc.

[2] A. C. Guyton, J. E. Hall (2006). Textbook of Medical Physiology. Philadelphia, Saunders.

[3] Institute, National Heart Blood and Lung. "What Causes a Heart Attack?" Heart Attack. Retrieved 19 Mar. 2009, from http://www.nhlbi.nih.gov/health/dci/Diseases/HeartAttack/HeartAttack_Causes.html.

[4] Krijnen, P. A., R. Nijmeijer, C. J. Meijer, C. A. Visser, C. E. Hack and H. W. Niessen (2002). "Apoptosis in myocardial ischaemia and infarction." J Clin Pathol **55**(11): 801-811.

[5] Boersma, H. H., B. L. Kietselaer, L. M. Stolk, A. Bennaghmouch, L. Hofstra, J. Narula, G. A. Heidendal and C. P. Reutelingsperger (2005). "Past, present, and future of annexin A5: from protein discovery to clinical applications." J Nucl Med **46**(12): 2035-2050.

[6] Zwaal, R. F. and A. J. Schroit (1997). "Pathophysiologic implications of membrane phospholipid asymmetry in blood cells." Blood **89**(4): 1121-1132.

[7] Fadok, V. A., D. R. Voelker, P. A. Campbell, J. J. Cohen, D. L. Bratton and P. M. Henson (1992). "Exposure of phosphatidylserine on the surface of apoptotic lymphocytes triggers specific recognition and removal by macrophages." J Immunol **148**(7): 2207-2216.

[8] Emoto, K., N. Toyama-Sorimachi, H. Karasuyama, K. Inoue and M. Umeda (1997). "Exposure of phosphatidylethanolamine on the surface of apoptotic cells." Exp Cell Res **232**(2): 430-434.

[9] Wolters, S. L., M. F. Corsten, C. P. Reutelingsperger, J. Narula and L. Hofstra (2007). "Cardiovascular molecular imaging of apoptosis." Eur J Nucl Med Mol Imaging **34 Suppl 1**: S86-98.

[10] Agency, U. S. Environmental Protection. "Technetium-99." Radiation Protection. Retrieved 18 May 2010, from http://www.epa.gov/rpdweb00/radionuclides/technetium.html.

[11] Arano, Y. (2002). "Recent advances in 99mTc radiopharmaceuticals." Ann Nucl Med **16**(2): 79-93.

[12] Koopman, G., C. P. Reutelingsperger, G. A. Kuijten, R. M. Keehnen, S. T. Pals and M. H. van Oers (1994). "Annexin V for flow cytometric detection of phosphatidylserine expression on B cells undergoing apoptosis." Blood **84**(5): 1415-1420.

[13] Kietselaer, B. L., C. P. Reutelingsperger, H. H. Boersma, G. A. Heidendal, I. H. Liem, H. J. Crijns, J. Narula and L. Hofstra (2007). "Noninvasive detection of programmed cell loss with 99mTc-labeled annexin A5 in heart failure." J Nucl Med **48**(4): 562-567.

[14] Zhao, M., Z. Li and S. Bugenhagen (2008). "99mTc-labeled duramycin as a novel phosphatidylethanolamine-binding molecular probe." J Nucl Med **49**(8): 1345-1352.

[15] Taki, J., T. Higuchi, A. Kawashima, J. F. Tait, S. Kinuya, A. Muramori, I. Matsunari, K. Nakajima, N. Tonami and H. W. Strauss (2004). "Detection of cardiomyocyte death in a rat model of ischemia and reperfusion using 99mTc-labeled annexin V." J Nucl Med **45**(9): 1536-1541.

[16] Zhao, M., D. A. Beauregard, L. Loizou, B. Davletov and K. M. Brindle (2001). "Non-invasive detection of apoptosis using magnetic resonance imaging and a targeted contrast agent." Nat Med **7**(11): 1241-1244.

[17] Zhao, M., X. Zhu, S. Ji, J. Zhou, K. S. Ozker, W. Fang, R. C. Molthen and R. S. Hellman (2006). "99mTc-labeled C2A domain of synaptotagmin I as a target-specific molecular probe for noninvasive imaging of acute myocardial infarction." J Nucl Med **47**(8): 1367-1374.

[18] Audi, S., M. Poellmann, X. Zhu, Z. Li and M. Zhao (2007). "Quantitative analysis of [99mTc]C2A-GST distribution in the area at risk after myocardial ischemia and reperfusion using a compartmental model." Nucl Med Biol **34**(8): 897-905.

[19] Wheeldon, L. W., Z. Schumert and D. A. Turner (1965). "Lipid composition of heart muscle homogenate." J Lipid Res **6**(4): 481-489.

[20] Verhoven, B., R. A. Schlegel and P. Williamson (1995). "Mechanisms of phosphatidylserine exposure, a phagocyte recognition signal, on apoptotic T lymphocytes." J Exp Med **182**(5): 1597-1601.

[21] Iwamoto, K., T. Hayakawa, M. Murate, A. Makino, K. Ito, T. Fujisawa and T. Kobayashi (2007). "Curvature-dependent recognition of ethanolamine phospholipids by duramycin and cinnamycin." Biophys J **93**(5): 1608-1619.

[22] MathWorks, The. "ode45, ode23, ode113, ode15s, ode23s, ode23t, ode23tb." MATLAB Function Reference. Retrieved 5 May 2010, from http://www.mathworks.com/access/helpdesk_r13/help/techdoc/ref/ode15s.html.

[23] MathWorks, The. "lsqcurvefit." Optimization Toolbox. Retrieved 7 Jul. 2009, from http://www.mathworks.com/access/helpdesk/help/toolbox/optim/index.html?/access/helpdesk/help/toolbox/optim/&http://www.mathworks.com/access/helpdesk/help/helpdesk.html.

[24] Masugata, H., K. Mizushige, S. Senda, A. Kinoshita, H. Sakamoto, S. Sakamoto and H. Matsuo (1999). "Relationship between myocardial tissue density measured by microgravimetry and sound speed measured by acoustic microscopy." Ultrasound Med Biol **25**(9): 1459-1463.

[25] D.M. Bates, D.G. Watts (1988). Nonlinear Regression Analysis and its Applications. Hoboken, Wiley.

[26] Olivetti, G., J. M. Capasso, E. H. Sonnenblick and P. Anversa (1990). "Side-to-side slippage of myocytes participates in ventricular wall remodeling acutely after myocardial infarction in rats." Circ Res **67**(1): 23-34.

[27] Rubart, M. and L. J. Field (2006). "Cardiac regeneration: repopulating the heart." Annu Rev Physiol **68**: 29-49.

[28] Ran, S., A. Downes and P. E. Thorpe (2002). "Increased exposure of anionic phospholipids on the surface of tumor blood vessels." Cancer Res **62**(21): 6132-6140.

[29] Srivastava, R and KN Khanna (2009). "Stokes-Einstein relation in two- and three-dimensional fluids." J Chem Eng Data **54**(5): 1452-1456.

[30] Garcia, R. A., C. E. Forde and H. A. Godwin (2000). "Calcium triggers an intramolecular association of the C2 domains in synaptotagmin." Proc Natl Acad Sci U S A **97**(11): 5883-5888.

[31] Jensen, R. B., K. Lykke-Andersen, G. I. Frandsen, H. B. Nielsen, J. Haseloff, H. M. Jespersen, J. Mundy and K. Skriver (2000). "Promiscuous and specific phospholipid binding by domains in ZAC, a membrane-associated Arabidopsis protein with an ARF GAP zinc finger and a C2 domain." Plant Mol Biol **44**(6): 799-814.

[32] Brotz, H., G. Bierbaum, A. Markus, E. Molitor and H. G. Sahl (1995). "Mode of action of the lantibiotic mersacidin: inhibition of peptidoglycan biosynthesis via a novel mechanism?" Antimicrob Agents Chemother **39**(3): 714-719.

[33] H. Iida, S. Eberl, K. Kim, Y. Tamura, Y. Ono, M. Nakazawa, A. Sohlberg, T. Zeniya, T. Hayashi, H. Watabe (2008). "Absolute quantitation of myocardial blood flow with 201Tl and dynamic SPECT in canine: optimisation and validation of kinetic modelling." Eur J Nucl Med Mol Imaging **35**: 896–905.

APPENDIX A: GLOSSARY

B_0 –total number of available binding sites for the probe in vascular and tissue spaces (in nmol/g) (Model 2)

B_{01} –total number of binding sites in the vascular space (in nmol/g) (Model 3)

B_{02} –total number of binding sites in the tissue space (in nmol/g) (Model 3)

$[C_1]$ – concentration of unbound probe in the vascular space

$[C_2]$ –concentration of unbound probe in the tissue space

C_3 –amount of bound probe in the tissue space

C_4 –amount of bound probe in the vascular space

$C_b(t)$ –probe blood activity

$[C_{in}]$ –probe concentration in the blood that enters the coronary circulation

$C_{total}(t)$ –total amount of probe present in vascular and tissue spaces at time (t)

F – coronary circulation blood flow (1 ml/min/g)

ID – probe injected dose (g)

K_{PS} –rate of probe diffusion between the vascular and tissue spaces (in ml/min/g)

$K_S B_{0A}$ –rate of active probe sequestration in the tissue space (in ml/min/g) (Model 1)

$K_S B_{0I}$ –rate of partially inactive probe (^{99m}T-D) sequestration in the tissue space (in ml/min/g) (Model 1)

K_{SA} – binding rate constant of the active probe in vascular and/or tissue space (in ml/nmol/min/g) (Models 2 and 3)

K_{SI} –binding rate constant of the partially inactive probe in vascular and/or tissue space (in ml/nmol/min/g) (Models 2 and 3)

m –mass of myocardial infract tissue (g)

MW – probe molecular weight (g/mol)

V_1 – volume of vascular space of the myocardium (in ml/g)

V_2 – volume of tissue space of the myocardium (in ml/g)

APPENDIX B: MATLAB Code

Two Graphical User Interfaces (GUIs) have been developed: 1) to fit one of three compartmental models to probe uptake data in normal and infarct myocardium, and 2) to simulate the probe uptake in normal and infarct myocardium for a given set of model parameters. Both GUIs must be opened using MATLAB version 7.6.0 or later and all m-files and data files must be located in the current directory.

General Instructions for Model Optimization GUI

-Type **"GUI_opt"** in the command window and press enter

Viable Tissue Model

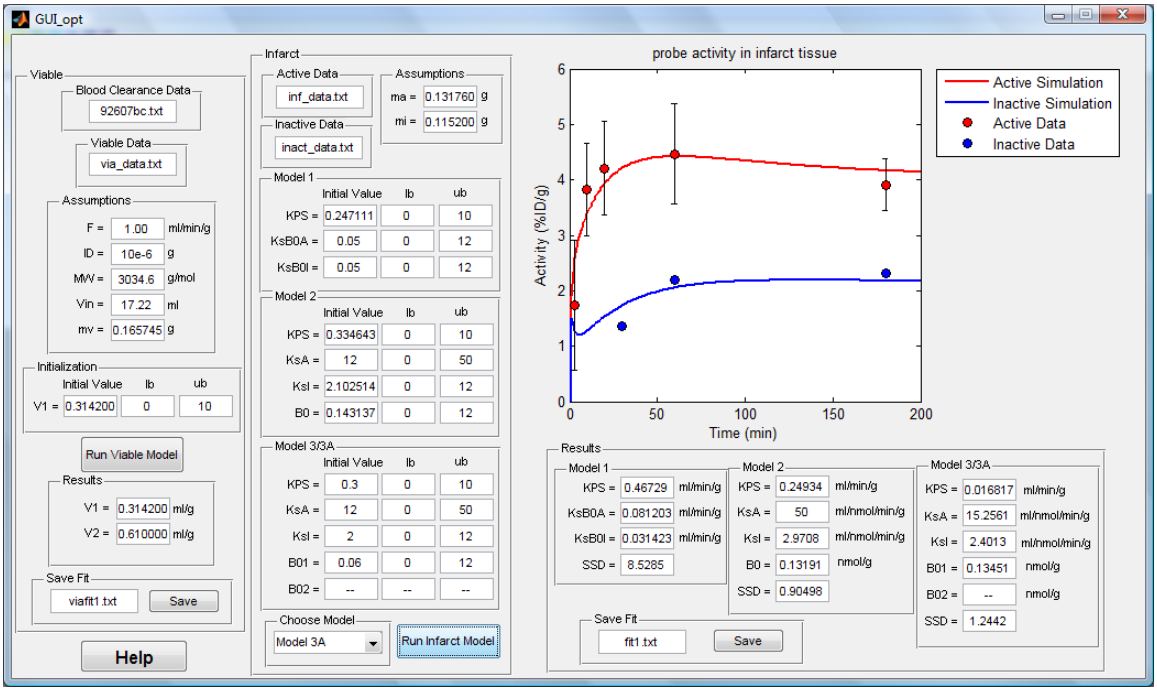
-Provide the filenames for probe blood clearance and probe uptake data in viable tissue
-Set values for the parameters whose values are assumed to be known
-Set initial value, lower bound, and upper bound for V_1
-Click "Run Viable Model"
    -Estimation for V_1 and resulting V_2 are presented in "Results" panel
    -Model fits and uptake data are plotted
-Define filename and click "Save" to save viable model fit to data

Infarct Tissue Model

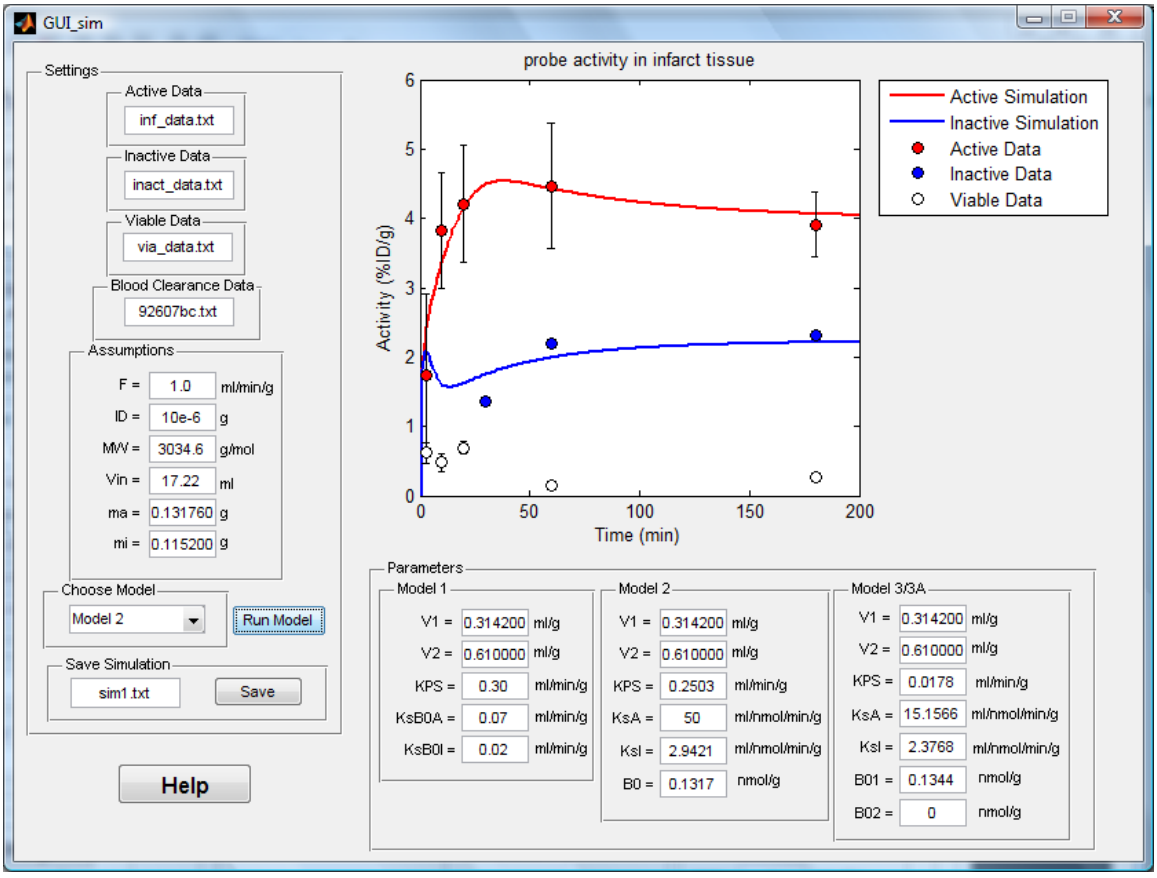
-Provide filenames for the active and inactive probe uptake data in infract tissue
-Set values for the parameters whose values are assumed to be known
-Set initial value, lower bound, and upper bound for all parameters in the model you will run
-Choose which model you want to fit to your data from the pull-down menu
-Click "Run Infarct Model"
    -Estimations for model parameters and SSD are presented in "Results" panel
    -Model fits and uptake data are plotted
-Define filename and click "Save" to save model fit to probe uptake data in infarct tissue

General Instructions for Model Simulation GUI

-Type **"GUI_sim"** in the command window and press enter
-Provide filenames for probe uptake in viable tissue, active probe uptake in infarct tissue, and inactive probe uptake in infarct tissue
-[Provide the filename for the probe blood clearance data
-Set values for the model parameters whose values are assumed to be known
-Provide values for the other model parameters
-Choose which model you want to simulate from the pull-down menu
-Click "Run Model"
    -Model simulation and uptake data are plotted
-Define filename and click "Save" to save model simulation



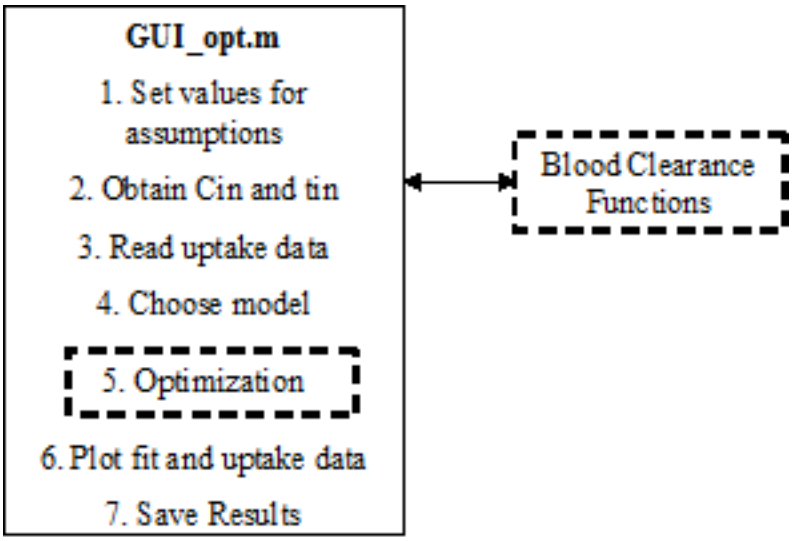
Model Optimization GUI



Model Simulation GUI

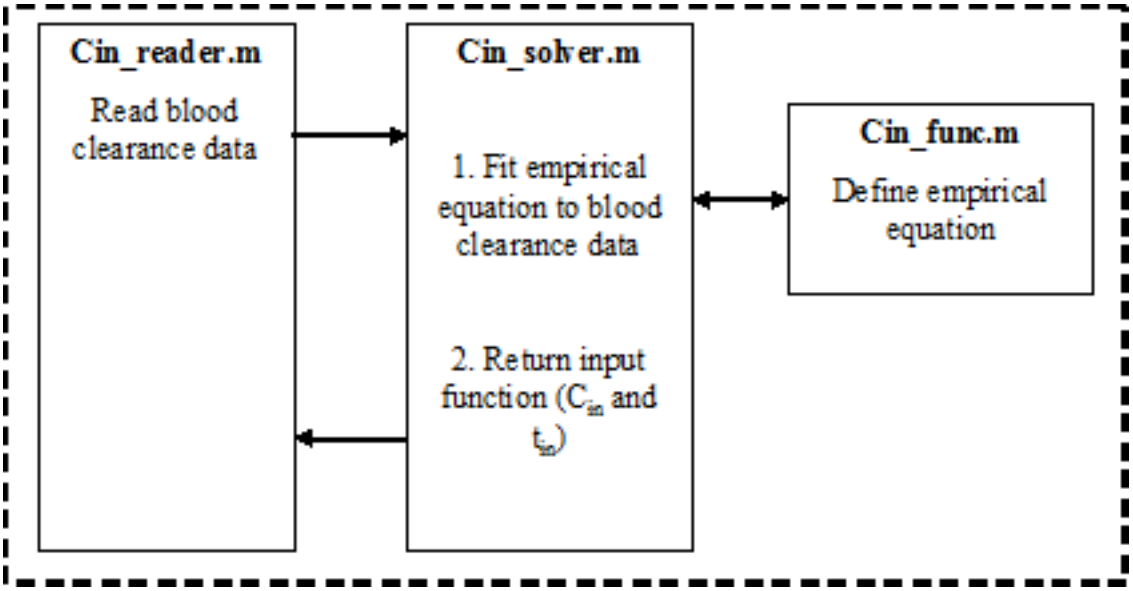
General Description of Model Optimization Code

The main m-file is **"GUI_opt.m"** which defines the GUI itself and reads input from the user. First, the values are set for themodel parameters whose values are assumed to be known (F, ID, MW, Vin, and m). Next, C_{in} and t_{in} (which define the input function to the model) are obtained from a set of functions related to the blood clearance.



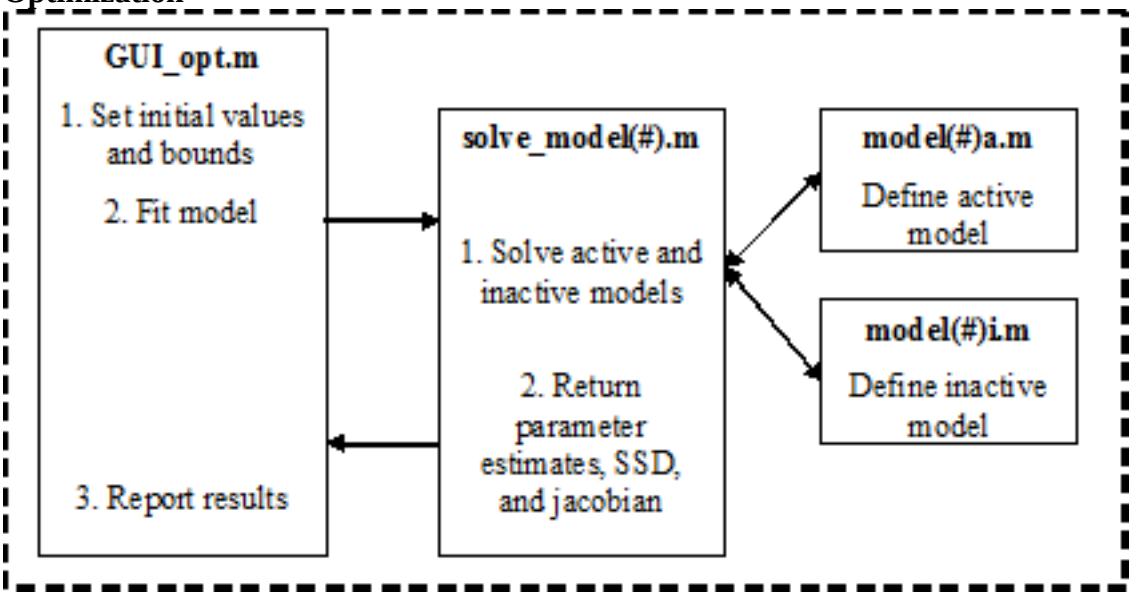
"Cin_reader.m" first reads the blood clearance data from a file defined by the user. This data is passed to **"Cin_solver.m"** which fits an empirical equation to the blood clearance data. The empirical equation is defined by **"Cin_func.m"**. **"Cin_solver.m"** then returns C_{in} and t_{in} which define the input function.

Blood Clearance Functions



"GUI_opt.m" next reads the uptake data files defined by the user. The optimization protocol then continues based on which model the user has selected from the pull-down menu. **"solve_model(#).m"**, **"model(#)a.m"**, and **"model(#)i.m"** are defined for each of the models.

Optimization



First, the initial values and bounds for the parameters are set as defined by the user. Next, **"solve_model(#).m"** is called to solve both the active and inactive models simultaneously. **"model(#)a.m"**, and **"model(#)i.m"** define the models for the active and inactive probe, respectively. **"solve_model(#).m"** returns the estimated parameter values along with the SSD and Jacobian matrix of the model fit to **"GUI_opt.m"**.

"GUI_opt.m" derives a correlation matrix and measures of precision of the parameter estimation from the Jacobian. **"GUI_opt.m"** then reports the parameter estimation results to the GUI before plotting the model fit and uptake data. The user is then given the option to save the model fit to a text file in the current directory.

The estimation of V_1 from the viable tissue probe uptake data is carried out in a similar manner to the protocol described above. The model simulation protocol is structured similarly, calling the blood clearance functions and model definitions from **"GUI_sim.m"**.

All m-files are presented on the following pages beginning with the main functions (**"GUI_opt.m"** and **"GUI_sim.m"**), then the blood clearance functions (**"Cin_reader.m"**, **"Cin_solver.m"**, and **"Cin_func.m"**), and finally the optimization functions (**"solve_model1.m"**, **"solve_model2.m"**, **"solve_model3.m"**, **"solve_model3A.m"**, **"solve_modelv.m"**, **"model1a.m"**, **"model1i.m"**, **"model2a.m"**, **"model2i.m"**, **"model3a.m"**, **"model3i.m"**, **"model3Aa.m"**, **"model3Ai.m"**, and **"modelv.m"**).

```
function varargout = GUI_opt(varargin)
% GUI_OPT M-file for GUI_opt.fig
%      GUI_OPT, by itself, creates a new GUI_OPT or raises the
existing
%      singleton*.
%
%      H = GUI_OPT returns the handle to a new GUI_OPT or the
handle to
%      the existing singleton*.
%
%      GUI_OPT('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in GUI_OPT.M with the given
input arguments.
%
%      GUI_OPT('Property','Value',...) creates a new GUI_OPT or
raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before GUI_opt_OpeningFunction gets
called.  An
%      unrecognized property name or invalid value makes
property application
%      stop.  All inputs are passed to GUI_opt_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI
allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help GUI_opt

% Last Modified by GUIDE v2.5 08-Jun-2010 16:21:06

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

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


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

handles.model = 1;

% Choose default command line output for GUI_opt
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

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


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

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

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

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


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

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



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

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


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

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



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

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


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

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


% --- Executes on selection change in choosemodel_pu.
function choosemodel_pu_Callback(hObject, eventdata, handles)
```

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

% Hints: contents = get(hObject,'String') returns
choosemodel_pu contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from choosemodel_pu

val=get(hObject,'Value');
str=get(hObject,'String');
switch str{val}
    case 'Model 1'
        handles.model = 1;
    case 'Model 2'
        handles.model = 2;
    case 'Model 3'
        handles.model = 3;
    case 'Model 3A'
        handles.model = 3.1;
        set(handles.M3_B02_in_edit,'String','--');
        set(handles.M3_B02_lb_edit,'String','--');
        set(handles.M3_B02_ub_edit,'String','--');
end
guidata(hObject, handles);

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

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


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

%Read data
global BC_data act_data inact_data
BC_data = get(handles.BCdata_edit,'String');
act_data = get(handles.actdata_edit,'String');
inact_data = get(handles.inactdata_edit,'String');

global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step tt
ttt

%Assumptions:
F = str2double(get(handles.F_edit,'String'));
ID = str2double(get(handles.ID_edit,'String'));
MW = str2double(get(handles.MW_edit,'String'));
Vin = str2double(get(handles.Vin_edit,'String'));
ma = str2double(get(handles.ma_edit,'String'));
mi = str2double(get(handles.mi_edit,'String'));
V1  = str2double(get(handles.V1_edit,'String'));
V2  = str2double(get(handles.V2_edit,'String'));

step = 1;

%obtain Cin and tin
[Cin,tin] = Cin_reader;


%-------------------------------------------------------------
------------
%open uptake data

    %open active file
    fid = fopen(act_data);
    %if file doesn't exist...
    while fid < 1
        infile = input('invalid filename, please reenter:
','s');
        fid = fopen(infile);
    end
    %scan file - data is a cell of floating point arrays
    data = textscan(fid,'%f %f %f');
    fclose(fid);
    %data has 3 columns
    %1:time, 2:uptake data, 3:standard deviation
```

```
    tt = (data{1})';
    inf = (data{2})';
    inf_stdev = (data{3})';

    %open inactive file
    fid = fopen(inact_data);
    %if file doesn't exist...
    while fid < 1
        infile = input('invalid filename, please reenter:
','s');
        fid = fopen(infile);
    end
    %scan file - data is a cell of floating point arrays
    data = textscan(fid,'%f %f');
    fclose(fid);
    %data has 2 columns
    %1:time, 2:uptake data
    ttt = (data{1})';
    inact = (data{2})';

    %combine data to fit simulatneously
    comb_t    = [  tt  ttt  ];
    comb_data = [ inf inact ];

%-----------------------------------------------------------
% Model 1 -----------------------------------------------
if handles.model == 1
    %Set initial values
    p0(1) = str2double(get(handles.M1_KPS_in_edit,'String'));
    p0(2) = str2double(get(handles.M1_KsB0A_in_edit,'String'));
    p0(3) = str2double(get(handles.M1_KsB0I_in_edit,'String'));
    %set upper and lower bounds
    lb(1) = str2double(get(handles.M1_KPS_lb_edit,'String'));
    lb(2) = str2double(get(handles.M1_KsB0A_lb_edit,'String'));
    lb(3) = str2double(get(handles.M1_KsB0I_lb_edit,'String'));
    ub(1) = str2double(get(handles.M1_KPS_ub_edit,'String'));
    ub(2) = str2double(get(handles.M1_KsB0A_ub_edit,'String'));
    ub(3) = str2double(get(handles.M1_KsB0I_ub_edit,'String'));

    %solve by least squares.  solve_model1.m defines the
parameters to be
    %solved by model1a.m and model1i.m.
    %Numbers below used to track number of iterations
    fprintf('\nfitting probe
data...\n.........1.........2.........3.........4.........5....
.....6.........7.........8.........9.........0........\n');
    [p,SSD_model,residual,exitflag,output,lambda,jacobian] =
lsqcurvefit(@solve_model1,p0,comb_t,comb_data,lb,ub);
    %returns parameters in array 'p'

    % correlation matrix
    [Q,R] = qr(jacobian);
    NP = length(p0);
    R1 = R(1:NP,:);
    h = inv(R1'*R1);
    for i = 1:NP;
        for j = 1:NP;
            cc(i,j) = h(i,j)/((h(i,i)*h(j,j))^0.5);
        end
    end
    cc
    %confidence intervals
    s_2 = SSD_model/(length(comb_data) - NP+2);
    hh = inv(jacobian'*jacobian);
    hh = hh*s_2;
    for i = 1:NP;
        seb(i) = ((s_2)^0.5)*(h(i,i)^0.5);
        StdError(i) = (hh(i,i)^0.5);
    end
    tt_dis = 1.90;  %7df
    seb = seb.*tt_dis;
    seb
    %standard error
    StdError

    global  KPS KsB0A KsB0I
    %set parameter to variable names
       KPS = p(1);
     KsB0A = p(2);
     KsB0I = p(3);

     %report results
     set(handles.M1_KPS_res_edit,'String',num2str(KPS));
     set(handles.M1_KsB0A_res_edit,'String',num2str(KsB0A));
     set(handles.M1_KsB0I_res_edit,'String',num2str(KsB0I));
     set(handles.M1_SSD_edit,'String',num2str(SSD_model));

%-------------------------------------------------
%active solver

global Ctotal1 Ctotal2 T
t = 0 : step : 200;
%             func   t span  init C  no options parameters
```

```
[T,C] = ode15s(@model1a,t,  [0 0 0], [],KPS,KsB0A);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3) / ID *100 / ma; %ID/g


%------------------------------------------------------
%inactive solver

 %             func   t span  init C  no options parameters
[T,C] = ode15s(@model1i,t,  [0 0 0], [],KPS,KsB0I);

%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3) / ID *100 / mi; %ID/g

% coefficient of variation for total model
    summ2 = 0;
    for i = 1:length(comb_data)
        summ2 = summ2 + comb_data(i);
    end

    coeff_model = ((sqrt(SSD_model/(length(comb_data) -
3)))/(summ2/length(comb_data)))*100;
    fprintf('    coeff_model = %f\n',
coeff_model);  %cofficient of variation for total model
end

%-------------------------------------------------------------
% Model 2 -----------------------------------------------------
if handles.model == 2
    %set initial values
    p0(1) = str2double(get(handles.M2_KPS_in_edit,'String'));
    p0(2) = str2double(get(handles.M2_KsA_in_edit,'String'));
    p0(3) = str2double(get(handles.M2_KsI_in_edit,'String'));
    p0(4) = str2double(get(handles.M2_B0_in_edit,'String'));
    %set upper and lower bounds
    lb(1) = str2double(get(handles.M2_KPS_lb_edit,'String'));
    lb(2) = str2double(get(handles.M2_KsA_lb_edit,'String'));
    lb(3) = str2double(get(handles.M2_KsI_lb_edit,'String'));
    lb(4) = str2double(get(handles.M2_B0_lb_edit,'String'));
    ub(1) = str2double(get(handles.M2_KPS_ub_edit,'String'));
    ub(2) = str2double(get(handles.M2_KsA_ub_edit,'String'));
    ub(3) = str2double(get(handles.M2_KsI_ub_edit,'String'));
    ub(4) = str2double(get(handles.M2_B0_ub_edit,'String'));


 %solve by least squares.  solve_model2.m defines the
parameters to be
    %solved by model2a.m and model2i.m.
    %Numbers below used to track number of iterations
    fprintf('\nfitting probe
data...\n.........1.........2.........3.........4.........5....
.....6.........7.........8.........9.........0........\n');


    [p,SSD_model,residual,exitflag,output,lambda,jacobian] =
lsqcurvefit(@solve_model2,p0,comb_t,comb_data,lb,ub);
    %returns parameters in array 'p'


    [Q,R] = qr(jacobian);
    NP = length(p0);
    R1 = R(1:NP,:);
    h = inv(R1'*R1);
    for i = 1:NP;
        for j = 1:NP;
            cc(i,j) = h(i,j)/((h(i,i)*h(j,j))^0.5);
        end
    end
    cc
    %confidence intervals
    s_2 = SSD_model/(length(comb_data) - NP+2);
    hh = inv(jacobian'*jacobian);
    hh = hh*s_2;
    for i = 1:NP;
        seb(i) = ((s_2)^0.5)*(h(i,i)^0.5);
        StdError(i) = (hh(i,i)^0.5);
```

```
    end
    tt_dis = 1.94;     %6df, a=.05
    seb = seb.*tt_dis;
    seb
    StdError

    global KPS KsA KsI B0
    %set parameter to variable names
      KPS = p(1);
      KsA = p(2);
      KsI = p(3);
       B0 = p(4);

    %report results
     set(handles.M2_KPS_res_edit,'String',num2str(KPS));
     set(handles.M2_KsA_res_edit,'String',num2str(KsA));
     set(handles.M2_KsI_res_edit,'String',num2str(KsI));
     set(handles.M2_B0_res_edit,'String',num2str(B0));
     set(handles.M2_SSD_edit,'String',num2str(SSD_model));

%-----------------------------------------------------
%active solver

global Ctotal1 Ctotal2 T
t = 0 : step : 200;
%             func   t span  init C  no options parameters
[T,C] = ode15s(@model2a,t,  [0 0 0], [],KPS,KsA,B0);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3) / ID *100 / ma; %ID/g

%-----------------------------------------------------
%inactive solver

 %             func   t span  init C  no options parameters
[T,C] = ode15s(@model2i,t,  [0 0 0], [],KPS,KsI,B0);

%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3) / ID *100 / mi; %ID/g



% coefficient of variation for total model
    summ2 = 0;
    for i = 1:length(comb_data)
        summ2 = summ2 + comb_data(i);
    end

    coeff_model = ((sqrt(SSD_model/(length(comb_data) -
3)))/(summ2/length(comb_data)))*100;
    fprintf('    coeff_model = %f\n',
coeff_model);  %cofficient of variation for total model
end

%-----------------------------------------------------------
% Model 3 ----------------------------------------------
if handles.model == 3
    %set initial values
    p0(1) = str2double(get(handles.M3_KPS_in_edit,'String'));
    p0(2) = str2double(get(handles.M3_KsA_in_edit,'String'));
    p0(3) = str2double(get(handles.M3_KsI_in_edit,'String'));
    p0(4) = str2double(get(handles.M3_B01_in_edit,'String'));
    p0(5) = str2double(get(handles.M3_B02_in_edit,'String'));
    %set upper and lower bounds
    lb(1) = str2double(get(handles.M3_KPS_lb_edit,'String'));
    lb(2) = str2double(get(handles.M3_KsA_lb_edit,'String'));
    lb(3) = str2double(get(handles.M3_KsI_lb_edit,'String'));
    lb(4) = str2double(get(handles.M3_B01_lb_edit,'String'));
    lb(5) = str2double(get(handles.M3_B02_lb_edit,'String'));
    ub(1) = str2double(get(handles.M3_KPS_ub_edit,'String'));
    ub(2) = str2double(get(handles.M3_KsA_ub_edit,'String'));
    ub(3) = str2double(get(handles.M3_KsI_ub_edit,'String'));
    ub(4) = str2double(get(handles.M3_B01_ub_edit,'String'));
    ub(5) = str2double(get(handles.M3_B02_ub_edit,'String'));
```

```
 %solve by least squares.  solve_model3.m defines the
parameters to be
    %solved by model3a.m and model3i.m.
    %Numbers below used to track number of iterations
    fprintf('\nfitting probe
data...\n.........1.........2.........3.........4.........5....
.....6.........7.........8.........9.........0........\n');

    [p,SSD_model,residual,exitflag,output,lambda,jacobian] =
lsqcurvefit(@solve_model3,p0,comb_t,comb_data,lb,ub);
    %returns parameters in array 'p'


    [Q,R] = qr(jacobian);
    NP = length(p0);
    R1 = R(1:NP,:);
    h = inv(R1'*R1);
    for i = 1:NP;
        for j = 1:NP;
            cc(i,j) = h(i,j)/((h(i,i)*h(j,j))^0.5);
        end
    end
    cc
    %confidence intervals
    s_2 = SSD_model/(length(comb_data) - NP+2);
    hh = inv(jacobian'*jacobian);
    hh = hh*s_2;
    for i = 1:NP;
        seb(i) = ((s_2)^0.5)*(h(i,i)^0.5);
        StdError(i) = (hh(i,i)^0.5);
    end
    tt_dis = 2.02;     %5df, a=.05
    seb = seb.*tt_dis;
    seb
    %standard error
    StdError

    global KPS KsA KsI B01 B02
    %set parameter to variable names
      KPS = p(1);
      KsA = p(2);
      KsI = p(3);
      B01 = p(4);
      B02 = p(5);

     %report results
     set(handles.M3_KPS_res_edit,'String',num2str(KPS));
     set(handles.M3_KsA_res_edit,'String',num2str(KsA));
     set(handles.M3_KsI_res_edit,'String',num2str(KsI));
     set(handles.M3_B01_res_edit,'String',num2str(B01));
     set(handles.M3_B02_res_edit,'String',num2str(B02));
     set(handles.M3_SSD_edit,'String',num2str(SSD_model));

%-------------------------------------------------
%active solver

global Ctotal1 Ctotal2 T
t = 0 : step : 200;
%            func   t span  init C  no options parameters
[T,C] = ode15s(@model3a,t,  [0 0 0 0], [],KPS,KsA,B01,B02);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C3 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3 + C4) / ID *100 / ma; %ID/g


%-------------------------------------------------
%inactive solver

 %             func   t span  init C  no options parameters
[T,C] = ode15s(@model3i,t,  [0 0 0 0], [],KPS,KsI,B01,B02);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3 + C4) / ID *100 / mi; %ID/g
```

```
% coefficient of variation for total model
    summ2 = 0;
    for i = 1:length(comb_data)
        summ2 = summ2 + comb_data(i);
    end

    coeff_model = ((sqrt(SSD_model/(length(comb_data) -
3)))/(summ2/length(comb_data)))*100;
    fprintf('    coeff_model = %f\n',
coeff_model);  %cofficient of variation for total model
end

%-------------------------------------------------------------
% Model 3A ----------------------------------------------------
if handles.model == 3.1
    %set initial values
    p0(1) = str2double(get(handles.M3_KPS_in_edit,'String'));
    p0(2) = str2double(get(handles.M3_KsA_in_edit,'String'));
    p0(3) = str2double(get(handles.M3_KsI_in_edit,'String'));
    p0(4) = str2double(get(handles.M3_B01_in_edit,'String'));
    %set upper and lower bounds
    lb(1) = str2double(get(handles.M3_KPS_lb_edit,'String'));
    lb(2) = str2double(get(handles.M3_KsA_lb_edit,'String'));
    lb(3) = str2double(get(handles.M3_KsI_lb_edit,'String'));
    lb(4) = str2double(get(handles.M3_B01_lb_edit,'String'));
    ub(1) = str2double(get(handles.M3_KPS_ub_edit,'String'));
    ub(2) = str2double(get(handles.M3_KsA_ub_edit,'String'));
    ub(3) = str2double(get(handles.M3_KsI_ub_edit,'String'));
    ub(4) = str2double(get(handles.M3_B01_ub_edit,'String'));


 %solve by least squares.  solve_model3A.m defines the
parameters to be
    %solved by model3Aa.m and model3Ai.m.
    %Numbers below used to track number of iterations
    fprintf('\nfitting probe
data...\n.........1.........2.........3.........4.........5....
.....6.........7.........8.........9.........0........\n');

    [p,SSD_model,residual,exitflag,output,lambda,jacobian] =
lsqcurvefit(@solve_model3A,p0,comb_t,comb_data,lb,ub);
    %returns parameters in array 'p'

    %correlation matrix
    [Q,R] = qr(jacobian);
    NP = length(p0);
    R1 = R(1:NP,:);
    h = inv(R1'*R1);
    for i = 1:NP;
        for j = 1:NP;
            cc(i,j) = h(i,j)/((h(i,i)*h(j,j))^0.5);
        end
    end
    cc
    %confidence intervals
    s_2 = SSD_model/(length(comb_data) - NP+2);
    hh = inv(jacobian'*jacobian);
    hh = hh*s_2;
    for i = 1:NP;
        seb(i) = ((s_2)^0.5)*(h(i,i)^0.5);
        StdError(i) = (hh(i,i)^0.5);
    end
    tt_dis = 1.94;     %6df, a=.05
    seb = seb.*tt_dis;
    seb
    %standard error
    StdError

    global KPS KsA KsI B01
    %set parameter to variable names
    KPS = p(1);
    KsA = p(2);
    KsI = p(3);
    B01 = p(4);

    %report results
     set(handles.M3_KPS_res_edit,'String',num2str(KPS));
     set(handles.M3_KsA_res_edit,'String',num2str(KsA));
     set(handles.M3_KsI_res_edit,'String',num2str(KsI));
     set(handles.M3_B01_res_edit,'String',num2str(B01));
     set(handles.M3_B02_res_edit,'String','--');
     set(handles.M3_SSD_edit,'String',num2str(SSD_model));

%-------------------------------------------------
%active solver

global Ctotal1 Ctotal2 T
t = 0 : step : 200;
%             func   t span  init C  no options parameters
[T,C] = ode15s(@model3Aa,t,  [0 0 0 0], [],KPS,KsA,B01,0);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
```

```
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3 + C4) / ID *100 / ma; %ID/g


%-------------------------------------------------
%inactive solver

 %             func   t span  init C  no options parameters
[T,C] = ode15s(@model3Ai,t,  [0 0 0 0], [],KPS,KsI,B01,0);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3 + C4) / ID *100 / mi; %ID/g


% coefficient of variation for total model
    summ2 = 0;
    for i = 1:length(comb_data)
        summ2 = summ2 + comb_data(i);
    end

    coeff_model = ((sqrt(SSD_model/(length(comb_data) -
3)))/(summ2/length(comb_data)))*100;
    fprintf('    coeff_model = %f\n',
coeff_model);  %cofficient of variation for total model

end

%plot uptake data and fits
axes(handles.uptake_plot);
plot(T,Ctotal1,'r-','LineWidth',2);
hold on;
plot(T,Ctotal2,'b-','LineWidth',2);
errorbar(tt,inf,inf_stdev,'ko','MarkerEdgeColor','k','MarkerFac
eColor','r');
plot(ttt,inact,'ko','MarkerEdgeColor','k','MarkerFaceColor','b'
);

%figure properties
legend('Active Simulation','Inactive Simulation','Active
Data','Inactive Data');
legend('Location','NorthEastOutside');
xlabel('Time (min)'); ylabel('Activity (%ID/g)');
title('probe activity in infarct tissue');

hold off



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

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


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

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

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

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


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

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



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

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


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

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



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

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


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

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

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

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


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

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



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

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


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

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



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

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


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

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

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

%Read data
global BC_data via_data
BC_data = get(handles.BCdata_edit,'String');
via_data = get(handles.viadata_edit,'String');

global F ID MW Vin V1 V2 mv a b alpha1 alpha2 alpha3 step

%Assumptions:
F = str2double(get(handles.F_edit,'String'));
ID = str2double(get(handles.ID_edit,'String'));
MW = str2double(get(handles.MW_edit,'String'));
Vin = str2double(get(handles.Vin_edit,'String'));
mv = str2double(get(handles.mv_edit,'String'));

step = 1;

%obtain Cin and tin
[Cin,tin] = Cin_reader;

%check parameters
fprintf('\nBlood Activity Parameters:\n');
fprintf('------------------------------\n');
fprintf('        a = %f\n', a);
fprintf('        b = %f\n', b);
fprintf('   alpha1 = %f\n', alpha1);
fprintf('   alpha2 = %f\n', alpha2);
fprintf('   alpha3 = %f\n', alpha3);

%-----------------------------------------------------------------
------------
%open uptake data

    %open file
    fid = fopen(via_data);
    %if file doesn't exist...
    while fid < 1
        infile = input('invalid filename, please reenter:
','s');
        fid = fopen(infile);
    end
    %scan file - data is a cell of floating point arrays
    data = textscan(fid,'%f %f %f');
    fclose(fid);
    %data has 3 columns
    %1:time, 2:uptake data, 3:standard deviation
    tvia = (data{1})';
    via = (data{2})';
    via_stdev = (data{3})';


    %print assumptions
    fprintf('\nAssumptions: \n');
    fprintf('------------------------------\n');
    fprintf('      F = %f ml/min/g\n', F);
    fprintf('     ID = %f g\n', ID);
    fprintf('     MW = %f g/mol\n', MW);
    fprintf('    Vin = %f ml\n',Vin);
    fprintf('     mv = %f g\n', mv);

    %read initial settings
    v0 = str2double(get(handles.V1_in_edit,'String'));
    vlb = str2double(get(handles.V1_lb_edit,'String'));
    vub = str2double(get(handles.V1_ub_edit,'String'));


    %viable fit
    [v,SSD_via,residual,exitflag,output,lambda,jacobian] =
lsqcurvefit(@solve_modelv,v0,tvia,via,vlb,vub);

    [Q,R] = qr(jacobian);
    NP = length(v0);
    R1 = R(1:NP,:);
    h = inv(R1'*R1);
    for i = 1:NP;
        for j = 1:NP;
            cc(i,j) = h(i,j)/((h(i,i)*h(j,j))^0.5);
        end
    end
    cc
    %confidence intervals
    s_2 = SSD_via/(length(via) - NP+1);
    hh = inv(jacobian'*jacobian);
    hh = hh*s_2;
```

```
    for i = 1:NP;
        seb(i) = ((s_2)^0.5)*(h(i,i)^0.5);
        StdError(i) = (hh(i,i)^0.5);
    end
    tt_dis = 2.02;     %5df, a=.05
    seb = seb.*tt_dis;
    seb
    StdError

    global V1
    V1 = v(1)
    V2 = 0.9142 - V1;
    SSD_via

    %report results
    set(handles.V1_edit,'String',num2str(V1));
    set(handles.V2_edit,'String',num2str(V2));


%-----------------------------------------------------
%-----------------------------------------------------
%viable solver

global Ctotalv T
t = 0 : step : 200;
    %             func   t span  init C  no options parameters
 [T,C] = ode15s(@modelv,t,  [0], [],V1);

    %C(1) = probe concentration, compartment 1

    %C1 is in terms of concentration (nmol/ml)
    %convert to %ID/g before charting
    C1 = (C(:,1) * (V1*mv) * 1e-9 * MW); %g
    Ctotalv = (C1) / ID *100 / mv; %ID/g


%-----------------------------------------------------------------
------------
    %plot data and fit
    axes(handles.uptake_plot);
    plot(T,Ctotalv,'k-','LineWidth',2);
    hold on

errorbar(tvia,via,via_stdev,'ko','MarkerEdgeColor','k','MarkerF
aceColor','w');

    %figure properties
    legend('viable simulation','viable data');
    legend('Location','BestOutside');
    xlabel('time (min)'); ylabel('activity (%ID/g)');
    title('probe activity in infarct tissue');

    hold off


function V1_edit_Callback(hObject, eventdata, handles)
% hObject    handle to V1_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of V1_edit as
text
%        str2double(get(hObject,'String')) returns contents of
V1_edit as a double


% --- Executes during object creation, after setting all
properties.
function V1_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to V1_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function V2_edit_Callback(hObject, eventdata, handles)
% hObject    handle to V2_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of V2_edit as
text
%        str2double(get(hObject,'String')) returns contents of
V2_edit as a double


% --- Executes during object creation, after setting all
properties.
function V2_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to V2_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



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

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


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

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



function M1_KPS_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KPS_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KPS_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KPS_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KPS_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KPS_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0A_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KsB0A_in_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M1_KsB0A_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0A_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0A_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KsB0A_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0I_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KsB0I_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KsB0I_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0I_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0I_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KsB0I_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KPS_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KPS_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KPS_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KPS_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KPS_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KPS_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0A_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KsB0A_lb_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M1_KsB0A_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0A_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0A_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KsB0A_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0I_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KsB0I_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KsB0I_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0I_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0I_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KsB0I_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KPS_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KPS_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KPS_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KPS_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KPS_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KPS_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0A_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KsB0A_ub_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M1_KsB0A_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0A_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0A_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KsB0A_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0I_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KsB0I_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KsB0I_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0I_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0I_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KsB0I_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_B01_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_B01_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_B01_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_B01_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B01_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B01_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_B01_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_B01_lb_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M3_B01_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_B01_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B01_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B01_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_B01_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_B01_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_B01_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_B01_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B01_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B01_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KPS_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KPS_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_KPS_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KPS_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KPS_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KPS_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsA_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KsA_in_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M3_KsA_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsA_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsA_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsA_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsI_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KsI_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_KsI_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsI_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsI_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsI_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KPS_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KPS_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_KPS_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KPS_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KPS_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KPS_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsA_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KsA_lb_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M3_KsA_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsA_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsA_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsA_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsI_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KsI_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_KsI_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsI_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsI_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsI_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KPS_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KPS_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_KPS_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KPS_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KPS_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KPS_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsA_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KsA_ub_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M3_KsA_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsA_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsA_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsA_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsI_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KsI_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_KsI_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsI_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsI_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsI_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsI_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsI_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_KsI_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsI_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsI_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsI_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsA_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsA_ub_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M2_KsA_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsA_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsA_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsA_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KPS_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KPS_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_KPS_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KPS_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KPS_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KPS_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsI_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsI_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_KsI_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsI_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsI_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsI_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsA_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsA_lb_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M2_KsA_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsA_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsA_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsA_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KPS_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KPS_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_KPS_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KPS_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KPS_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KPS_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsI_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsI_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_KsI_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsI_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsI_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsI_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsA_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsA_in_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M2_KsA_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsA_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsA_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsA_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KPS_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KPS_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_KPS_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KPS_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KPS_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KPS_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_B0_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_B0_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_B0_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_B0_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_B0_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_B0_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_B0_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_B0_lb_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M2_B0_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_B0_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_B0_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_B0_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_B0_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_B0_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_B0_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_B0_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_B0_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_B0_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_B02_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_B02_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_B02_in_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_B02_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B02_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B02_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_B02_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_B02_lb_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M3_B02_lb_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_B02_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B02_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B02_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_B02_ub_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_B02_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_B02_ub_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_B02_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B02_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B02_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_B01_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_B01_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_B01_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_B01_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B01_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B01_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KPS_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KPS_res_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M3_KPS_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KPS_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KPS_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KPS_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsA_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KsA_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_KsA_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsA_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsA_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsA_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsI_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KsI_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M3_KsI_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsI_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsI_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsI_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_B02_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_B02_res_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M3_B02_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M3_B02_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B02_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B02_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_B0_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_B0_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_B0_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_B0_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_B0_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_B0_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KPS_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KPS_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_KPS_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KPS_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KPS_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KPS_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsA_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsA_res_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M2_KsA_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsA_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsA_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsA_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsI_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsI_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M2_KsI_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsI_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsI_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsI_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0I_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KsB0I_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KsB0I_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0I_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0I_res_edit_CreateFcn(hObject, eventdata,
handles)
% hObject    handle to M1_KsB0I_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0A_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KsB0A_res_edit (see GCBO)
```

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

% Hints: get(hObject,'String') returns contents of
M1_KsB0A_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0A_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0A_res_edit_CreateFcn(hObject, eventdata,
handles)
% hObject    handle to M1_KsB0A_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KPS_res_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KPS_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KPS_res_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KPS_res_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KPS_res_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KPS_res_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_SSD_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_SSD_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M2_SSD_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M2_SSD_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_SSD_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_SSD_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_SSD_edit_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to M1_SSD_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M1_SSD_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M1_SSD_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_SSD_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_SSD_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function V1_in_edit_Callback(hObject, eventdata, handles)
% hObject    handle to V1_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of V1_in_edit
as text
%        str2double(get(hObject,'String')) returns contents of
V1_in_edit as a double


% --- Executes during object creation, after setting all
properties.
function V1_in_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to V1_in_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function V1_lb_edit_Callback(hObject, eventdata, handles)
% hObject    handle to V1_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of V1_lb_edit
as text
%        str2double(get(hObject,'String')) returns contents of
V1_lb_edit as a double


% --- Executes during object creation, after setting all
properties.
function V1_lb_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to V1_lb_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function V1_ub_edit_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to V1_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of V1_ub_edit
as text
%        str2double(get(hObject,'String')) returns contents of
V1_ub_edit as a double


% --- Executes during object creation, after setting all
properties.
function V1_ub_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to V1_ub_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_SSD_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_SSD_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M3_SSD_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M3_SSD_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_SSD_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_SSD_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



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

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


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

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

```
% --- Executes on button press in save_pb.
function save_pb_Callback(hObject, eventdata, handles)
% hObject    handle to save_pb (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
    global Ctotal1 Ctotal2 T
    %get filename from GUI
    fname = get(handles.fname_edit,'String');
    %form filename, open file
    outfile = [ fname ];

    fid = fopen(outfile,'w');
    fprintf('Output file saved as %s\n',outfile);

    %print output file
    for i = 1:length(Ctotal1)

fprintf(fid,'%f\t%f\t%f\r\n',T(i),Ctotal1(i),Ctotal2(i));
    end
    fclose(fid);



% --- Executes on button press in viasave_pb.
function viasave_pb_Callback(hObject, eventdata, handles)
% hObject    handle to viasave_pb (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
    global Ctotalv T
    %get filename from GUI
    fname = get(handles.viafname_edit,'String');
    %form filename, open file
    outfile = [ fname ];

    fid = fopen(outfile,'w');
    fprintf('Output file saved as %s\n',outfile);

    %print output file
    for i = 1:length(Ctotalv)
        fprintf(fid,'%f\t%f\r\n',T(i),Ctotalv(i));
    end
    fclose(fid);


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

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


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

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



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

open('GUI_help.pdf');


function varargout = GUI_sim(varargin)
% GUI_SIM M-file for GUI_sim.fig
%      GUI_SIM, by itself, creates a new GUI_SIM or raises the
existing
%      singleton*.
%
```

```
%      H = GUI_SIM returns the handle to a new GUI_SIM or the
handle to
%      the existing singleton*.
%
%      GUI_SIM('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in GUI_SIM.M with the given
input arguments.
%
%      GUI_SIM('Property','Value',...) creates a new GUI_SIM or
raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before GUI_opt_OpeningFunction gets
called.  An
%      unrecognized property name or invalid value makes
property application
%      stop.  All inputs are passed to GUI_sim_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI
allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help GUI_sim

% Last Modified by GUIDE v2.5 08-Jun-2010 16:26:49

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

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


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

handles.model = 1;

% Choose default command line output for GUI_sim
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

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


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

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



function BCdata_edit_Callback(hObject, eventdata, handles)
% hObject    handle to BCdata_edit (see GCBO)
```

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

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


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

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



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

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


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

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



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

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


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

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

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

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


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

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



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

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


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

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



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

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


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

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



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

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


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

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



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

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


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

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



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

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


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

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



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

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


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

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

```
function M1_V1_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_V1_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M1_V1_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M1_V1_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_V1_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_V1_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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

```
function M1_V2_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_V2_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M1_V2_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M1_V2_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_V2_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_V2_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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

```
function M3_B01_edit_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to M3_B01_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M3_B01_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M3_B01_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B01_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B01_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KPS_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KPS_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M3_KPS_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M3_KPS_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KPS_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KPS_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsA_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_KsA_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M3_KsA_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsA_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsA_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsA_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_KsI_edit_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to M3_KsI_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M3_KsI_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M3_KsI_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_KsI_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_KsI_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_B02_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_B02_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M3_B02_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M3_B02_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_B02_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_B02_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_B0_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_B0_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M2_B0_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M2_B0_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_B0_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_B0_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KPS_edit_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to M2_KPS_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M2_KPS_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M2_KPS_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KPS_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KPS_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsA_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsA_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M2_KsA_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsA_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsA_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsA_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_KsI_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_KsI_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M2_KsI_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M2_KsI_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_KsI_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_KsI_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0I_edit_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to M1_KsB0I_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KsB0I_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0I_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0I_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KsB0I_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KsB0A_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KsB0A_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of
M1_KsB0A_edit as text
%        str2double(get(hObject,'String')) returns contents of
M1_KsB0A_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KsB0A_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KsB0A_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M1_KPS_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M1_KPS_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M1_KPS_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M1_KPS_edit as a double


% --- Executes during object creation, after setting all
properties.
function M1_KPS_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M1_KPS_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_V1_edit_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to M3_V1_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M3_V1_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M3_V1_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_V1_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_V1_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M3_V2_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M3_V2_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M3_V2_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M3_V2_edit as a double


% --- Executes during object creation, after setting all
properties.
function M3_V2_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M3_V2_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_V1_edit_Callback(hObject, eventdata, handles)
% hObject    handle to M2_V1_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M2_V1_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M2_V1_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_V1_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_V1_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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



function M2_V2_edit_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to M2_V2_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of M2_V2_edit
as text
%        str2double(get(hObject,'String')) returns contents of
M2_V2_edit as a double


% --- Executes during object creation, after setting all
properties.
function M2_V2_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to M2_V2_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

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


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

% Hints: contents = get(hObject,'String') returns
choosemodel_pu contents as cell array
%        contents{get(hObject,'Value')} returns selected item
from choosemodel_pu

val=get(hObject,'Value');
str=get(hObject,'String');
switch str{val}
    case 'Model 1'
        handles.model = 1;
    case 'Model 2'
        handles.model = 2;
    case 'Model 3'
        handles.model = 3;
    case 'Model 3A'
        handles.model = 3.1;
        set(handles.M3_B02_edit,'String','--');
end
guidata(hObject, handles);

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

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


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

%Read data
global BC_data act_data inact_data via_data
BC_data = get(handles.BCdata_edit,'String');
act_data = get(handles.actdata_edit,'String');
inact_data = get(handles.inactdata_edit,'String');
via_data = get(handles.viadata_edit,'String');

global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step tt
ttt

%Assumptions:
F = str2double(get(handles.F_edit,'String'));
ID = str2double(get(handles.ID_edit,'String'));
```

```
MW = str2double(get(handles.MW_edit,'String'));
Vin = str2double(get(handles.Vin_edit,'String'));
ma = str2double(get(handles.ma_edit,'String'));
mi = str2double(get(handles.mi_edit,'String'));

step = 1;

%obtain Cin and tin
[Cin,tin] = Cin_reader;


%-------------------------------------------------------------
------------
%open uptake data

    %open active file
    fid = fopen(act_data);
    %if file doesn't exist...
    while fid < 1
        infile = input('invalid filename, please reenter:
','s');
        fid = fopen(infile);
    end
    %scan file - data is a cell of floating point arrays
    data = textscan(fid,'%f %f %f');
    fclose(fid);
    %data has 3 columns
    %1:time, 2:uptake data, 3:standard deviation
    tt = (data{1})';
    inf = (data{2})';
    inf_stdev = (data{3})';

    %open inactive file
    fid = fopen(inact_data);
    %if file doesn't exist...
    while fid < 1
        infile = input('invalid filename, please reenter:
','s');
        fid = fopen(infile);
    end
    %scan file - data is a cell of floating point arrays
    data = textscan(fid,'%f %f');
    fclose(fid);
    %data has 2 columns
    %1:time, 2:uptake data
    ttt = (data{1})';
    inact = (data{2})';

    %open viable file
    fid = fopen(via_data);
    %if file doesn't exist...
    while fid < 1
        infile = input('invalid filename, please reenter:
','s');
        fid = fopen(infile);
    end
    %scan file - data is a cell of floating point arrays
    data = textscan(fid,'%f %f %f');
    fclose(fid);
    %data has 3 columns
    %1:time, 2:uptake data
    tvia = (data{1})';
    via = (data{2})';
    via_stdev = (data{3})';

    %combine data to fit simulatneously
    comb_t    = [  tt  ttt  ];
    comb_data = [ inf inact ];

%-------------------------------------------------------------
% Model 1 -----------------------------------------------------
if handles.model == 1

    %Set parameter values
    V1  = str2double(get(handles.M1_V1_edit,'String'));
    V2  = str2double(get(handles.M1_V2_edit,'String'));
    KPS = str2double(get(handles.M1_KPS_edit,'String'));
  KsB0A = str2double(get(handles.M1_KsB0A_edit,'String'));
  KsB0I = str2double(get(handles.M1_KsB0I_edit,'String'));

%-------------------------------------------------------
%active solver

global Ctotal1 Ctotal2 T
t = 0 : step : 200;
%             func   t span  init C  no options parameters
[T,C] = ode15s(@model1a,t,  [0 0 0], [],KPS,KsB0A);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
```

```
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3) / ID *100 / ma; %ID/g


%-------------------------------------------------
%inactive solver

 %             func   t span  init C  no options parameters
[T,C] = ode15s(@model1i,t,  [0 0 0], [],KPS,KsB0I);

%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3) / ID *100 / mi; %ID/g

end

%-------------------------------------------------------------
% Model 2 -----------------------------------------------------
if handles.model == 2

    %Set parameter values
    V1  = str2double(get(handles.M2_V1_edit,'String'));
    V2  = str2double(get(handles.M2_V2_edit,'String'));
    KPS = str2double(get(handles.M2_KPS_edit,'String'));
    KsA = str2double(get(handles.M2_KsA_edit,'String'));
    KsI = str2double(get(handles.M2_KsI_edit,'String'));
     B0 = str2double(get(handles.M2_B0_edit,'String'));

%-------------------------------------------------
%active solver

global Ctotal1 Ctotal2 T
t = 0 : step : 200;
%             func   t span  init C  no options parameters
[T,C] = ode15s(@model2a,t,  [0 0 0], [],KPS,KsA,B0);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3) / ID *100 / ma; %ID/g

%-------------------------------------------------
%inactive solver

 %             func   t span  init C  no options parameters
[T,C] = ode15s(@model2i,t,  [0 0 0], [],KPS,KsI,B0);

%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3) / ID *100 / mi; %ID/g

end

%-------------------------------------------------------------
% Model 3 -----------------------------------------------------
if handles.model == 3

    %Set parameter values
    V1  = str2double(get(handles.M3_V1_edit,'String'));
    V2  = str2double(get(handles.M3_V2_edit,'String'));
    KPS = str2double(get(handles.M3_KPS_edit,'String'));
    KsA = str2double(get(handles.M3_KsA_edit,'String'));
    KsI = str2double(get(handles.M3_KsI_edit,'String'));
    B01 = str2double(get(handles.M3_B01_edit,'String'));
    B02 = str2double(get(handles.M3_B02_edit,'String'));



%-------------------------------------------------
%active solver
```

```
global Ctotal1 Ctotal2 T
t = 0 : step : 200;
%             func   t span  init C  no options parameters
[T,C] = ode15s(@model3a,t,  [0 0 0 0], [],KPS,KsA,B01,B02);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C3 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3 + C4) / ID *100 / ma; %ID/g


%-----------------------------------------------------
%inactive solver

 %             func   t span  init C  no options parameters
[T,C] = ode15s(@model3i,t,  [0 0 0 0], [],KPS,KsI,B01,B02);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3 + C4) / ID *100 / mi; %ID/g

end

%-----------------------------------------------------------
% Model 3A ---------------------------------------------------
if handles.model == 3.1

    %Set parameter values
    V1  = str2double(get(handles.M3_V1_edit,'String'));
    V2  = str2double(get(handles.M3_V2_edit,'String'));
    KPS = str2double(get(handles.M3_KPS_edit,'String'));
    KsA = str2double(get(handles.M3_KsA_edit,'String'));
    KsI = str2double(get(handles.M3_KsI_edit,'String'));
    B01 = str2double(get(handles.M3_B01_edit,'String'));


%-----------------------------------------------------
%active solver

global Ctotal1 Ctotal2 T
t = 0 : step : 200;
%             func   t span  init C  no options parameters
[T,C] = ode15s(@model3Aa,t,  [0 0 0 0], [],KPS,KsA,B01,0);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3 + C4) / ID *100 / ma; %ID/g


%-----------------------------------------------------
%inactive solver

 %             func   t span  init C  no options parameters
[T,C] = ode15s(@model3Ai,t,  [0 0 0 0], [],KPS,KsI,B01,0);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3 + C4) / ID *100 / mi; %ID/g
```

```
end

%plot uptake data and fits
axes(handles.uptake_plot);
plot(T,Ctotal1,'r-','LineWidth',2);
hold on;
plot(T,Ctotal2,'b-','LineWidth',2);
errorbar(tt,inf,inf_stdev,'ko','MarkerEdgeColor','k','MarkerFac
eColor','r');
plot(ttt,inact,'ko','MarkerEdgeColor','k','MarkerFaceColor','b'
);
errorbar(tvia,via,via_stdev,'ko','MarkerEdgeColor','k','MarkerF
aceColor','w');

%figure properties
legend('Active Simulation','Inactive Simulation','Active
Data','Inactive Data','Viable Data');
legend('Location','NorthEastOutside');
xlabel('Time (min)'); ylabel('Activity (%ID/g)');
title('probe activity in infarct tissue');

hold off
```

```
% --- Executes on button press in save_pb.
function save_pb_Callback(hObject, eventdata, handles)
% hObject    handle to save_pb (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
    global Ctotal1 Ctotal2 T
    %get filename from GUI
    fname = get(handles.fname_edit,'String');
    %form filename, open file
    outfile = [ fname ];

    fid = fopen(outfile,'w');
    fprintf('Output file saved as %s\n',outfile);

    %print output file
    for i = 1:length(Ctotal1)

fprintf(fid,'%f\t%f\t%f\r\n',T(i),Ctotal1(i),Ctotal2(i));
    end
    fclose(fid);


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

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


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

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

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

open('GUI_help.pdf');
```

```
function [Cin_fit,t_fit] = Cin_reader
%This function reads Cin from a text file provided by the user
%Passes the data to Cin_solver
%Outputs fit data to a text file
global F ID MW Vin m a b alpha1 alpha2 alpha3 BC_data

%prompt user for filename
%note that user must specify '.txt'
%if the file is not saved in the current directory, the full
directory path
%   must be provided
%files must be text files, tab delimited with time in left
column and
%   activity in right column
%input is not case-sensitive
fprintf('\n\n-------------------------------------------------
-----------\n');

%open file
fid = fopen(BC_data);

%if file doesn't exist...
while fid < 1
    infile = input('invalid filename, please reenter: ','s');
    fid = fopen(infile);
end

%scan file - data is a cell of floating point arrays
data = textscan(fid,'%f %f');
fclose(fid);


%assign time and activity
tinput = data{1}';
Cinput = data{2}';


%send to Cin solver function to get fit
[Cin_fit,t_fit] = Cin_solver(tinput,Cinput);

function [Cin_fit,t_fit] = Cin_solver(t_data,Cin_data)
%This function solves for Cin
%   User specifies intial parameter values
%   Cin is solved by least-squares fit...see Cin_func for
equation
%   Best-fit parameters are printed
%   Data and fit are charted
%   User is asked to accept fit (or reenter initial parameters)
%   Function returns Cin best fit and corresponding time vector
%   As part of a larger program, this function is called by
Cin_reader

global F ID MW Vin m a b alpha1 alpha2 alpha3
%Start with the following intial paramters,
%These should work well
%       a  b  alpha1 alpha2 alpha 3
p0 = [  20 30 1      .005      .05];



fprintf('-----------------------------------\n');

%-------------------------------------------------
%normalize blood activity data

%shift data if 100% is defined at t ~= 0
if t_data(1) ~= 0
    for i = 2:length(t_data)
        t_data(i) = t_data(i) - t_data(1);
    end
    t_data(1) = 0;
    fprintf('\nTime points shifted so that Cin(t=0) = 100\n\n');
end



%-------------------------------------------------
%calculate best-fit parameters


%see top for intial parameters

%set upper, lower bounds
lb = [   0   0  0  0  0 ];
ub = [ 100 100 10 10 10 ];


%Calculate parameters by least squares fit
fprintf('\nfitting blood activity data...\n');
[p, resnorm] = lsqcurvefit(@Cin_func, p0, t_data,
Cin_data,lb,ub);

%assign parameters to variables names
     a = p(1);  %amplitude
```

```
     b = p(2);  %amplitude
alpha1 = p(3);  %rate constant
alpha2 = p(4);  %rate constant
alpha3 = p(5);  %rate constant

%print parameters
fprintf('\nBlood Activity Parameters:\n');
fprintf('------------------------------\n');
fprintf('        a = %f\n', a);
fprintf('        b = %f\n', b);
fprintf('   alpha1 = %f\n', alpha1);
fprintf('   alpha2 = %f\n', alpha2);
fprintf('   alpha3 = %f\n', alpha3);


%Calculate final 60 minute fit from parameters
%   a high level of precision (.05 minutes) is appropriate for
calculating
%   half-life
t_fit60 = 0 : .05 : t_data(length(t_data)); %time vector
Cin_fit60 = Cin_func(p,t_fit60);             %fit

%Calculate 180 minute (extrapolated) fit from parameters
%   this lower-precision value will be returned from the
function and used
%   later in the compartmental model
t_fit = 0:1:180;
Cin_fit = Cin_func(p,t_fit);


%-------------------------------------------------
%runs test

%initialize array for splined fit
Cin_fit_spl = zeros(length(t_data),1);

%adjust model to fit original time series (spline)
Cin_fit_spl = spline(t_fit60,Cin_fit60,t_data);

%calculate residuals - will be array length of data
res = Cin_data - Cin_fit_spl;
%Find number of runs, positive residuals
runs = 1; posres = 0;
for i = 1 : length(t_data)-1
    if res(i) >0 && res(i+1) <0 || res(i) <0 && res(i+1) >0
        runs = runs + 1;
    end
    if res(i) > 0
        posres = posres + 1;
    end
end


%calculate expeceted number of runs
negres = length(res) - posres;  %calculate negative runs
expruns = (2*posres*negres) / (posres+negres) + 1;
expruns = int16(expruns);  %ensures number is an integer


%-------------------------------------------------
%print statistics

fprintf('\nBlood Activity Statistics:\n');
fprintf('--------------------------\n');
fprintf('     data = %i points\n',length(t_data));
fprintf('      SSD = %f\n', resnorm);
fprintf('     runs = %i runs\n', runs);
fprintf(' expected = %i runs\n', expruns);



%-------------------------------------------------
%calculate other values


%biologic half-life
i = 0; j = 1;
while i < 1
    if Cin_fit60(j) < 50
        fprintf(' halflife = %f min\n\n',t_fit60(j));
        i = 1;
    else
        j = j+1;
    end
end


%-------------------------------------------------

end

function Cin = Cin_func(p,t)
%This function describes Cin, the input probe concentration
%   Cin is in units of % activity
```

```
%This function is called by Cin_solver.m

%parameters
     a = p(1);  %amplitude
     b = p(2);  %amplitude
alpha1 = p(3);  %rate constant
alpha2 = p(4);  %rate constant
alpha3 = p(5);  %rate constant

%empirical equation to describe blood clearance
Cin = a * exp(-alpha1*t) + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);



function Cmatch = solve_model1(p,comb_t)
%numerical solver for active probe

%declare global variables
global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step tt
ttt


%define parameters
   KPS = p(1);
 KsB0A = p(2);
 KsB0I = p(3);

fprintf('.');      %used to count number of iterations
%-----------------------------------------------------
%active solver

t = 0 : step : 200;
%             func   t span  init C  no options parameters
[T,C] = ode15s(@model1a,t,  [0 0 0], [],KPS,KsB0A);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3) / ID *100 / ma; %ID/g

Cmatch1 = Ctotal1(int16(tt(1))/step+1);
for i = 2:length(tt)
    Cmatch1 = [ Cmatch1, Ctotal1(int16(tt(i))/step+1)];
end
%-----------------------------------------------------
%inactive solver

    %             func   t span  init C  no options parameters
 [T,C] = ode15s(@model1i,t,  [0 0 0], [],KPS,KsB0I);

%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3) / ID *100 / mi; %ID/g


Cmatch2 = Ctotal2(int16(ttt(1))/step+1);
for i = 2:length(ttt)
    Cmatch2 = [ Cmatch2, Ctotal2(int16(ttt(i))/step+1)];
end

Cmatch = [ Cmatch1 Cmatch2 ];

function Cmatch = solve_model2(p,comb_t)
%numerical solver for active probe

%declare global variables
global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step tt
ttt


%define parameters
  KPS = p(1);
  KsA = p(2);
  KsI = p(3);
   B0 = p(4);

fprintf('.');      %used to count number of iterations
%-----------------------------------------------------
%active solver
```

```
t = 0 : step : 200;
%            func   t span  init C  no options parameters
[T,C] = ode15s(@model2a,t,  [0 0 0], [],KPS,KsA,B0);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3) / ID *100 / ma; %ID/g

Cmatch1 = Ctotal1(int16(tt(1))/step+1);
for i = 2:length(tt)
    Cmatch1 = [ Cmatch1, Ctotal1(int16(tt(i))/step+1)];
end
%-----------------------------------------------------
%inactive solver

    %            func   t span  init C  no options parameters
 [T,C] = ode15s(@model2i,t,  [0 0 0], [],KPS,KsI,B0);

%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 is in terms of mass (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3) / ID *100 / mi; %ID/g


Cmatch2 = Ctotal2(int16(ttt(1))/step+1);
for i = 2:length(ttt)
    Cmatch2 = [ Cmatch2, Ctotal2(int16(ttt(i))/step+1)];
end

Cmatch = [ Cmatch1 Cmatch2 ];

function Cmatch = solve_model3(p,comb_t)

%numerical solver for active probe

%declare global variables
global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step tt
ttt


%define parameters
  KPS = p(1);
  KsA = p(2);
  KsI = p(3);
  B01 = p(4);
  B02 = p(5);

fprintf('.');      %used to count number of iterations
%-----------------------------------------------------
%active solver

t = 0 : step : 200;
%            func   t span  init C  no options parameters
[T,C] = ode15s(@model3a,t,  [0 0 0 0], [],KPS,KsA,B01,B02);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3 + C4) / ID *100 / ma; %ID/g

Cmatch1 = Ctotal1(int16(tt(1))/step+1);
for i = 2:length(tt)
    Cmatch1 = [ Cmatch1, Ctotal1(int16(tt(i))/step+1)];
end
%-----------------------------------------------------
%inactive solver

    %            func   t span  init C  no options parameters
[T,C] = ode15s(@model3i,t,  [0 0 0 0], [],KPS,KsI,B01,B02);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
```

```
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3 + C4) / ID *100 / mi; %ID/g


Cmatch2 = Ctotal2(int16(ttt(1))/step+1);
for i = 2:length(ttt)
    Cmatch2 = [ Cmatch2, Ctotal2(int16(ttt(i))/step+1)];
end

Cmatch = [ Cmatch1 Cmatch2 ];

function Cmatch = solve_model3A(p,comb_t)
%numerical solver for active probe
%this function is called by optimize_model.m

%declare global variables
global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step tt
ttt


%define parameters
  KPS = p(1);
  KsA = p(2);
  KsI = p(3);
  B01 = p(4);

fprintf('.');      %used to count number of iterations
%-----------------------------------------------------
%active solver

t = 0 : step : 200;
%             func   t span  init C  no options parameters
[T,C] = ode15s(@model3Aa,t,  [0 0 0 0], [],KPS,KsA,B01,0);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*ma) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*ma) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal1 = (C1 + C2 + C3 + C4) / ID *100 / ma; %ID/g

Cmatch1 = Ctotal1(int16(tt(1))/step+1);
for i = 2:length(tt)
    Cmatch1 = [ Cmatch1, Ctotal1(int16(tt(i))/step+1)];
end
%-----------------------------------------------------
%inactive solver

    %             func   t span  init C  no options parameters
[T,C] = ode15s(@model3Ai,t,  [0 0 0 0], [],KPS,KsI,B01,0);
%C(1) = probe concentration, compartment 1
%C(2) = probe concentration, compartment 2
%C(3) = probe amount, compartment 3
%C(4) = probe amount, compartment 4

%C1 and C2 are in terms of concentration (nmol/ml)
%C3 and C4 are in terms of amount (nmol)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mi) * 1e-9 * MW); %g
C2 = (C(:,2) * (V2*mi) * 1e-9 * MW); %g
C3 = (C(:,3)           * 1e-9 * MW); %g
C4 = (C(:,4)           * 1e-9 * MW); %g
Ctotal2 = (C1 + C2 + C3 + C4) / ID *100 / mi; %ID/g


Cmatch2 = Ctotal2(int16(ttt(1))/step+1);
for i = 2:length(ttt)
    Cmatch2 = [ Cmatch2, Ctotal2(int16(ttt(i))/step+1)];
end

Cmatch = [ Cmatch1 Cmatch2 ];

function Cmatch = solve_modelv(v,tvia)
%numerical solver for active probe
%this function is called by optimize_model.m

%declare global variables
global F ID MW Vin mv a b alpha1 alpha2 alpha3 step
```

```
%define parameters
   V1 = v(1);

fprintf('.');      %used to count number of iterations
%-------------------------------------------------
%viable solver

t = 0 : step : 200;
%             func   t span  init C  no options parameters
[T,C] = ode15s(@modelv,t,  [0], [],V1);

%C(1) = probe concentration, compartment 1

%C1 is in terms of concentration (nmol/ml)
%convert to %ID/g before charting
C1 = (C(:,1) * (V1*mv) * 1e-9 * MW); %g
Ctotalv = (C1) / ID *100 / mv; %ID/g

Cmatch1 = Ctotalv(int16(tvia(1))/step+1);
for i = 2:length(tvia)
    Cmatch1 = [ Cmatch1, Ctotalv(int16(tvia(i))/step+1)];
end


Cmatch = [Cmatch1];

function dC = model1a(t,C,KPS,KsB0)
%This function defines the three differential equations that
make up the
%   compartmental model.  It is called by other numerical
solving functions

global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step
%initialize array
dC = zeros(3,1);

%input function
C_in = a * exp(-alpha1*t)  + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);
C_in  = (( C_in *  ID /100 ) / MW ) * 1e9 / Vin; % nmol/ml (uM)
%note that C1 and C2 are concentrations (nmol/ml)
%          C3 is amount (nmol),

%equations
dC(1) = F/V1*(C_in - C(1)) + KPS/V1*(C(2) - C(1));
dC(2) = KPS/V2*(C(1) - C(2)) - KsB0/V2*C(2);
dC(3) = KsB0*ma*C(2);
```

```
function dC = model3a(t,C,KPS,KsB0)
%This function defines the three differential equations that
make up the
%   compartmental model.  It is called by other numerical
solving functions

global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step
%initialize array
dC = zeros(3,1);


%input function
C_in = a * exp(-alpha1*t)  + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);
C_in  = (( C_in *  ID /100 ) / MW ) * 1e9 / Vin; % nmol/ml (uM)
%note that C1 and C2 are concentrations (nmol/ml)
%          C3 is amount (nmol),

%equations
dC(1) = F/V1*(C_in - C(1)) + KPS/V1*(C(2) - C(1));
dC(2) = KPS/V2*(C(1) - C(2)) - KsB0/V2*C(2);
dC(3) = KsB0*mi*C(2);

function dC = model2a(t,C,KPS,Ks,B0)
%This function defines the three differential equations that
make up the
%   compartmental model.  It is called by other numerical
solving functions


global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step
%initialize array
dC = zeros(3,1);


%input function
C_in = a * exp(-alpha1*t)  + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);
C_in  = (( C_in *  ID /100 ) / MW ) * 1e9 / Vin; % nmol/ml (uM)
%note that C1 and C2 are concentrations (nmol/ml)
%          C3 is amount (nmol),

%equations
dC(1) = F/V1*(C_in - C(1)) + KPS/V1*(C(2) - C(1));
dC(2) = KPS/V2*(C(1) - C(2)) - Ks/V2*C(2)*(B0*ma - C(3));
dC(3) = Ks*ma*C(2)*(B0*ma - C(3));
```

```
function dC = model2i(t,C,KPS,Ks,B0)
%This function defines the three differential equations that
make up the
%   compartmental model.  It is called by other numerical
solving functions


global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step
%initialize array
dC = zeros(3,1);



%input function
C_in = a * exp(-alpha1*t)  + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);
C_in  = (( C_in *  ID /100 ) / MW ) * 1e9 / Vin; % nmol/ml (uM)
%note that C1 and C2 are concentrations (nmol/ml)
%          C3 is amount (nmol),

%equations
dC(1) = F/V1*(C_in - C(1)) + KPS/V1*(C(2) - C(1));
dC(2) = KPS/V2*(C(1) - C(2)) - Ks/V2*C(2)*(B0*mi - C(3));
dC(3) = Ks*mi*C(2)*(B0*mi - C(3));

function dC = model3a(t,C,KPS,Ks,B01,B02)
%This function defines the three differential equations that
make up the
%   compartmental model.  It is called by other numerical
solving functions

global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step
%initialize array
dC = zeros(4,1);


%input function
C_in = a * exp(-alpha1*t)  + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);
C_in  = (( C_in *  ID /100 ) / MW ) * 1e9 / Vin; % nmol/ml (uM)
%note that C1 and C2 are concentrations (nmol/ml)
%          C3 and C4 are amounts (nmol),

%equations
dC(1) = F/V1*(C_in - C(1)) + KPS/V1*(C(2) - C(1)) -
Ks/V1*C(1)*(B01*ma - C(4));
dC(2) = KPS/V2*(C(1) - C(2)) - Ks/V2*C(2)*(B02*ma - C(3));
dC(3) = Ks*ma*C(2)*(B02*ma - C(3));
dC(4) = Ks*ma*C(1)*(B01*ma - C(4));

function dC = model3i(t,C,KPS,Ks,B01,B02)
%This function defines the three differential equations that
make up the
%   compartmental model.  It is called by other numerical
solving functions

global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step
%initialize array
dC = zeros(4,1);

%input function
C_in = a * exp(-alpha1*t)  + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);
C_in  = (( C_in *  ID /100 ) / MW ) * 1e9 / Vin; % nmol/ml (uM)
%note that C1 and C2 are concentrations (nmol/ml)
%          C3 and C4 are amounts (nmol),

%equations
dC(1) = F/V1*(C_in - C(1)) + KPS/V1*(C(2) - C(1)) -
Ks/V1*C(1)*(B01*mi - C(4));
dC(2) = KPS/V2*(C(1) - C(2)) - Ks/V2*C(2)*(B02*mi - C(3));
dC(3) = Ks*mi*C(2)*(B02*mi - C(3));
dC(4) = Ks*mi*C(1)*(B01*mi - C(4));

function dC = model3Aa(t,C,KPS,Ks,B01,B02)
%This function defines the three differential equations that
make up the
%   compartmental model.  It is called by other numerical
solving functions


global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step
%initialize array
dC = zeros(4,1);

%input function
C_in = a * exp(-alpha1*t)  + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);
C_in  = (( C_in *  ID /100 ) / MW ) * 1e9 / Vin; % nmol/ml (uM)
%note that C1 and C2 are concentrations (nmol/ml)
%          C3 and C4 are amounts (nmol),

%equations
```

```
dC(1) = F/V1*(C_in - C(1)) + KPS/V1*(C(2) - C(1)) -
Ks/V1*C(1)*(B01*ma - C(4));
dC(2) = KPS/V2*(C(1) - C(2)) - Ks/V2*C(2)*(B02*ma - C(3));
dC(3) = Ks*ma*C(2)*(B02*ma - C(3));
dC(4) = Ks*ma*C(1)*(B01*ma - C(4));

function dC = model3Ai(t,C,KPS,Ks,B01,B02)
%This function defines the three differential equations that
make up the
%   compartmental model.  It is called by other numerical
solving functions

global F ID MW Vin V1 V2 ma mi a b alpha1 alpha2 alpha3 step
%initialize array
dC = zeros(4,1);

%input function
C_in = a * exp(-alpha1*t)  + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);
C_in  = (( C_in *  ID /100 ) / MW ) * 1e9 / Vin; % nmol/ml (uM)
%note that C1 and C2 are concentrations (nmol/ml)
%          C3 and C4 are amounts (nmol),

%equations
dC(1) = F/V1*(C_in - C(1)) + KPS/V1*(C(2) - C(1)) -
Ks/V1*C(1)*(B01*mi - C(4));
dC(2) = KPS/V2*(C(1) - C(2)) - Ks/V2*C(2)*(B02*mi - C(3));
dC(3) = Ks*mi*C(2)*(B02*mi - C(3));
dC(4) = Ks*mi*C(1)*(B01*mi - C(4));

function dC = model3v(t,C,V1)
%This function defines the three differential equations that
make up the
%   compartmental model.  It is called by other numerical
solving functions

global F ID MW Vin mv a b alpha1 alpha2 alpha3 step
%initialize array
dC = zeros(1,1);


%input function
C_in = a * exp(-alpha1*t)  + b * exp(-alpha2*t) + (100-a-b) *
exp(-alpha3*t);
C_in  = (( C_in *  ID /100 ) / MW ) * 1e9 / Vin; % nmol/ml (uM)
%note that C1 and C2 are concentrations (nmol/ml)
%          C3 and C4 are amounts (nmol),

%new equations
dC(1) = F/V1*(C_in - C(1));
```