Kinetic Analysis of dynamic MP4A PET Scans of Human Brain using Voxel based Nonlinear Least Squares Fitting

Inaugural - Dissertation

zur

Erlangung des Doktorgrades

der Mathematisch-Naturwissenschaftlichen Fakultät

der Universität zu Köln

vorgelegt von

Christoph Hohmann

aus Leverkusen

Hundt Druck GmbH, Köln
2009

Berichterstatter/in: Prof. Dr. Rainer Schrader

Prof. Dr. Ulrich Trottenberg

Tag der letzten mündlichen Prüfung: 16. 10. 2009

Kurzzusammenfassung

Dynamisches PET (Positronenemissionstomographie) mit verschiedenen Radiotracern wird als bildgebendes Verfahren eingesetzt zur in vivo Bestimmung verschiedener biochemischer Parameter im menschlichen Gehirn, wie z.B. Gesamtstoffwechselrate oder bestimmte Rezeptorkonzentrationen und Enzymaktivitäten. ^{11}C markiertes Methyl-4-Piperidyl Acetat (MP4A) und -Propionat (MP4P) werden als Radiotracer zur Aktivitätsbestimmung von Acetylcholinesterase (AChE) verwendet. Das Enzym dient als Indikator der Funktionalität des cholinergen Subsystems. Zur kinetischen Analyse dynamischer MP4A PET Messungen ohne Verwendung arterieller Blutdaten wird ein referenzbasiertes, irreversibles Tracermodell eingesetzt. Die Auswertung damit kann auf regionaler oder Voxelebene erfolgen, im zweiten Fall führt sie zu Bildern des die AChE-Aktivität anzeigenden Parameters k_3.

Die vorliegende Arbeit enthält eine Implementierung der voxelbasierten kinetischen Analyse mittels Kurvenanpassung im Sinne gewichteter kleinster Quadrate (NLS), die schnell genug ist für Standard PCs. Der gesamte Bildverarbeitungsprozess einschliesslich Normalisierung und Bewegungskorrektur, der von rekonstruierten PET Scans zu parametrischen k_3 Bildern und regionalen Mittelwerten führt, wurde automatisiert. Dabei kommen neue Techniken der Bildvorverarbeitung, ohne Verwendung starrer Masken, zum Einsatz.

Ein Schwerpunkt der Arbeit ist die Fehlerabschätzung von k_3 auf Voxel- und regionaler Ebene. Eine Formel für die voxelweise Standardabweichung wird hergeleitet. Sie basiert auf den Abweichungsquadraten und wird gegen Simulationsergebnisse validiert. Als größte Fehlerquelle für regionale Mittelwerte von k_3 stellten sich die Referenzkurven heraus. Hier konnten wesentliche Verbesserungen erzielt werden durch Verwendung adaptiver Putamenmasken und Erhöhung des Referenzvolumens von 5,4 auf 12,5 bis 16 ml. Außerdem wird eine Methode zur Korrektur der Referenzkurven im Falle nichtidealer Referenzregionen vorgestellt.

Für das verbesserte Verfahren wurden die Standardabweichungen regionaler Mittelwerte von k_3 mit Hilfe der PET Scans von 12 Probanden, die auf Sinogrammebene in je zwei disjunkte Datensätze zerlegt wurden, grob bestimmt. Danach ergibt sich ein absoluter Fehler von 0.0012 für typische Cortexregionen und 0,0053 für Hippocampus als Folge des Rauschens der voxelbasierten Aktivitätskurven, während die Referenzkurven für Fehler von ca. 0,0025 bzw. 0,0050 verantwortlich sind. Systematische Fehler verschiedener Art wurden mit Simulationen untersucht, ihr kombinierter Effekt liegt bei weniger als 3 Prozent des gemessenen k_3.

Das Verfahren wurde als Modul des Softwarepaketes VINCI verfügbar gemacht und im Rahmen klinischer Studien eingesetzt, zur Untersuchung der Parkinsonschen Krankheit und der Alzheimer Demenz.

Abstract

Dynamic PET (Positron Emission Tomography) involving a number of radiotracers is an established technique for in vivo estimation of biochemical parameters in human brain, such as the overall metabolic rate and certain receptor concentrations or enzyme activities. ^{11}C labeled methyl-4-piperidyl acetate (MP4A) and -propionate (MP4P) are established radiotracers for measuring activity of acetylcholine esterase (AChE), which relates to functionality of the cholinergic system. MP4A kinetic analysis without arterial blood sampling employs a reference tissue based "irreversible tracer model". Implementations can be region or voxel based, in the second case providing parametric images of k_3 which is an indicator of AChE activity.

This work introduces an implementation of voxel based kinetic analysis using weighted Nonlinear Least Squares fitting (NLS), which is fast enough for standard PCs. The entire workflow leading from reconstructed PET scans to parametric images of k_3, including normalization and correction for patient movement, has been automatized. Image preprocessing has been redefined and fixed masks are no longer required.

A focus of this work is error estimation of k_3 at the voxel and regional level. A formula is derived for voxel based estimation of random error, it is based on residual weighted squared differences and has been successfully validated against simulated data. The reference curves turned out to be the main source of errors in regional mean values of k_3. Major improvements were reached in this area by switching from fixed to adaptive Putamen masks and raising their volume from 5.4 to 12.5 or 16 ml. Also, a method for correcting reference curves obtained from nonideal reference tissues is presented.

For the improved implementation, random error of the mean k_3 of a number of cerebral regions has been assessed based on PET studies of 12 human subjects, by splitting them in two independent data sets at the sinogram level. According to this sample, absolute standard errors of 0.0012 in most cortex regions and 0.0053 in Hippocampus are induced by noise of voxel based activity curves, while errors of approximately 0.0025 and 0.0050 are induced by noise of the reference curves. Different types of systematic as well as noise-induced bias have been investigated by simulations; their combined effect on the computed k_3 was found below 3 percent.

The implementation is available as a modul of the VINCI software package and has been used in clinical studies on Parkinson's Disease and Alzheimer Dementia.

Acknowledgements

I thank Univ. Prof. Dr. Wolf-Dieter Heiss for the opportunity to work on this thesis and for his encouragement and generous financial support during my time as a research assistant. To Univ. Prof. Dr. Yves von Cramon I owe thanks for supporting this project and allowing me to use the resources of the Max-Planck Institute for Neurological Research, Cologne. I am very grateful to Univ. Prof. Dr. Karl Herholz (Wolfson Centre for Molecular Imaging, Manchester) for including me in his Alzheimer research group, suggesting the subject, allowing me to meet other renowned scientists through him, and many fruitful discussions.

I am very grateful to my mathematical supervisor, Univ. Prof. Dr. Rainer Schrader, for introducing me to the Max-Planck-Institute, supporting this work as a member of the math. nat. faculty of the University of Cologne, and for the outstanding support, encouragement and valuable advice he provided along the way.

I am deeply indebted to Dr. Stefan Vollmar, head of the IT group at MPIFNF, for allowing me to contribute to the VINCI project, introducing me to the Qt framework and other state-of-the-art software technologies, facilitating my integration into the MPIFNF community, his extraordinary help with this work and his constant encouragement and moral support along the way.

Very special thanks go to Michael Sué, chief developer of VINCI. Time and again, he went out of his way sacrificing many hours to solve software related and other problems and get me back on track. This work would not have been possible without him.

The same is true of Roman Krais who is in charge of the PET tomographs. He modified the acquisition protocols to make chapter 14 possible and participated in assembling the final “abba” images. He helped design the phantom experiment of chapter 7, and, together with Alexander Schuster, showed me how to remove a critical roadblock in reading the ECAT7 file format. Thank you very much for your role in making this work possible, and for providing much valuable background information on PET.

Many thanks to Alexander Schuster for his contribution to the software for adding sinograms, to Roswitha Rusniak who conducted the phantom experiment, and to Carmen Selbach for providing image materials.

To Dr. Cathleen Haense I owe thanks for the wonderful time we shared as colleagues in the Alzheimer project. Many thanks to her and Dr. Carsten Eggers for their feedback on the MP4A plugin.

I owe special gratitude to Prof. Dr. Klaus Wienhard and Dr. Heiko Backes for proofreading the final draft along with Dr. Stefan Vollmar, Michael Sué and Roman Krais, and providing many helpful suggestions.

To the staff of MPIFNF I owe thanks for their help along the way, especially Ingo Alt, Hans Boenner, Peter Spiegelberg and Daniel Rotter for their

technical support, and Andreas Huesgen for helping me with the Python language. And I want to thank all my colleagues and fellows at MPIFNF for the time we spent together and the inspiration they provided.

Last not least, I wish to express my gratitude to Dr. Roland Ulrich for his patience during the three years that we shared the same bureau, and for conversations and insights that will shape my memory of this time.

Contents

Chapter 1

Introduction

1.1 PET

Positron **E**mission **T**omography[1] is an imaging technique used in nuclear medicine. It produces 3-dimensional images of the distribution of a radioactive isotope in the body.

Nuclear imaging started in 1957 with the development of the gamma camera. It gives two dimensional low resolution images of radiotracer distribution. The radiotracer (commonly just called **tracer**) is a molecule containing a gamma emitting nuclide.

Based on the gamma camera, SPECT[2] was developed later as a 3-dimensional imaging technique. To localize the origin of gamma photons, collimators are used. They absorb more than 99 percent of the photons, thus sacrificing sensitivity and still being confined to very low resolutions.

Resolution and scanner sensitivity improved in 1976 with the arrival of PET. The idea is to use only such isotopes that emit pairs of gamma photons. More precisely, the photon pairs are the result of an annihilation event of an electron with its antiparticle, a positron, resulting in two gamma photons whose energy equals the mass of an electron, 511 keV, and whose divergence angle is close to 180 degrees. If both photons are detected, it happens simultaneously and can therefore be attributed to a single annihilation event on the connecting line of the two detectors **("Line of Response, LOR")**. This obliterates the need for collimators and also provides for better resolution. Another decisive advantage is that PET allows to correct for **attenuation**[3], thus enabling quantitative measurement of specific activity.

The positrons are emitted from the nuclide in a process called β^+ decay. The most frequently used β^+ emitting isotopes are listed in **Table 1.1**.

[1] This section is based on [16, 24, 25, 42, 43]

[2] SPECT=Single Photon Emission Computed Tomography

[3] loss of photons caused by interaction with tissue

Isotope	half life[min]	maximum range in H_2O [mm]
^{11}C	20.4	4.1
^{13}N	9.96	5.4
^{15}O	2.05	8.2
^{18}F	109.7	2.4

Table 1.1: *Isotopes used in PET [43]*

1.1.1 PET Scanners and Image Reconstruction

During **acquisition** of PET images, the patient is placed on a bed inside a scanner. Its main part is a ring of detectors. They contain crystals of a translucent scintillating material containing heavy elements, such as lutetium oxyorthosilicate (LSO). Detection of a gamma photon results in a flash of visible light which is amplified and recorded by a photomultiplier. Only "**coincidences**" are counted: the detection of two photons in different detectors during a time window of about 10 nanoseconds. Every such event is associated with a Line of Response (LOR). For each LOR, coincidences are counted over total scantime, the resulting dataset is called a **sinogram**. By the **reconstruction software**, the sinograms are processed to 3D images of specific activity (in units of Becquerel per ml).

1.1.2 Applications

PET allows for quantitative measurement of nuclide concentration. The tracer dose needed is orders of magnitude below pharmacological concentrations. Therefore almost every molecule, however toxic, is eligible as a radiotracer, if only it contains a C,N,O or F atom and can be synthesized within short time from a nonradioactive precursor. The most frequently used PET tracer is 2-^{18}F-desoxyglucose, commonly called **FDG**. Like ordinary glucose, it is taken up from the blood by every cell. It then undergoes phosphorylation as the first step of glycolysis, but cannot be further metabolized and therefore stays trapped in the cell. FDG has proven useful for a number of applications including brain scans. For the latter, it is particularly well suited since brain tissue feeds exclusively on glucose.

Another well-known tracer is ^{15}O labeled water, which is used to obtain perfusion images (section 9.2) of the brain. Its short half life makes it suitable for multiple application to a patient, allowing to localize functional centers in the brain by having the patients perform "tasks" during the scans.

Generally, PET is the method of choice for all types of "molecular imaging", as it is capable to localize molecules inside the body. Its specificity in doing so is unmatched by other methods including functional MR. Its application range grows with every new radiotracer developed.

1.1.3 Economic Aspects

For all isotopes of Table 1.1 except ^{18}F, half life is too short for transporting the tracer between production and scanning sites. This implies that ^{11}C, ^{13}N and ^{15}O tracers can only be used by institutes that have their own production facilities. They include a cyclotron for production of the nuclide and a team of radiochemists for incorporating it in the tracer molecule. Not many clinical centers can afford the expense.

The situation is somewhat better for ^{18}F. Given its half life of 110 minutes, the tracer must still be used at the day of production, but there remain a few hours for distribution within larger metropolitan areas. This explains why fluorine tracers like FDG found clinical application, while carbon tracers like MP4A and others are limited to scientific research at the few places that have the resources for making them.

1.1.4 Performance Limitations

The amount of radioactivity allowed for medical examinations is obviously limited. In our own study (section 2.11) the total number of decays taking place in the patient's body was about 10^{12}. About 4 percent occur in the brain (section A.5), and about 3 percent of these lead to coincidences in the PET scanner. The remaining 97 percent are lost for different reasons:

- a photon gets scattered in the patient's body
- a photon misses the detectors
- a photon fails to trigger an event in the detector crystal, or
- the event is not recorded during detector dead time

In addition, not every coincidence necessarily corresponds to a decay in its LOR. It can happen that

- a scattered photon and its partner are both detected, sparking a coincidence event on a "false" LOR
- detection of two unrelated photons occurs within the time window, leading to a "random coincidence".

These factors limit the amount of information from which images can be computed. Another set of limitations affects image resolution more directly:

- The positron is emitted with enough energy to travel a distance across the tissue prior to annihilation. The annihilation site and the site of decay can therefore be a few millimeters apart (see the column "maximum range in H_2O" in Table 1.1).

- As a consequence of the positron's residual speed upon annihilation, the divergence angle may differ up to 0.25 degrees from 180^o.
- The LORs are at least as thick as the detector crystals,
- some are even thicker since the crystals do not point in their direction [42]. So the length of the crystals contributes (by projection) to the width of LOR. The length is a few centimeters, this much is needed to achieve sufficient sensitivity.

1.1.4.1 Noise

All information gathered by the scanner consists in a number of discrete events (counted coincidences) with associated time and incomplete spatial information (events cannot be mapped to points in space, instead we have the LORs). On today's most advanced scanners, the whole information fits in a **listmode file** of 5 to 30 Gigabytes.

Events arise from radioactive decay, a random process. So they are affected by so-called **"shot noise"** from the very start. It propagates through image reconstruction and makes itself noticed in the images. Thus for principal reasons, PET can never reach resolutions comparable with magnetic resonance imaging (MRI) or computed X-ray tomography (CT). Images look "noisy" to the human eye and hardly reveal more than the crudest anatomical detail (see Figure 10.1, upper row).

1.1.4.2 Partial Volume Effects (PVE)

For all the reasons given in sections 1.1.4 and 1.1.4.1, a point source of radioactivity maps to a blurred spot on PET images, determined by a "point-spread function" looking much like a 3-dimensional normal distribution. Regional mean values of the reconstructed image will thus be corrupted by signal from the neighborhood, especially if the region in question is small or longitudinal: for instance, when averaging image intensity over the voxels of a Hippocampus mask, one may end up with 70% of the signal actually stemming from the patient's Hippocampus, and 30% from its surroundings. The problem is aggravated by any smoothing employed during image preprocessing. An example from this work: the reference curves C_r are less affected by PVE (section 1.1.4.2) than regional results because the former are sampled before the Gauss filtering step.

1.1.4.3 Absolute Quantification

of specific radioactivity in PET is achieved with a relative error in the range of 5 to 10 percent [24] by correcting the measured signal for attenuation. Attenuation assessment employs an extra "transmission scan" which is recorded before tracer injection, using an external source of radiation. For kinetic analysis (section 1.5) this type of error is irrelevant since it is the same in every frame (section 1.1.5).

1.1.5 Dynamic PET

is used to study changes in tracer concentration over time. Over a period of a few half lives after injection, several PET images (called **frames**) are shot and reconstructed independently. Needless to say that, as we subdivide the information, images become even noisier. Thus for every spatial region or voxel, we have an array of values. Plotting them against time gives a so called **Time Activity Curve**[4] (**TAC**), which can be subject to kinetic analysis.

TACs are only meaningful if all their values refer to the same voxel or region of the brain. To ensure this, the patient must either remain motionless during scantime, or the data must later be corrected for patient movement. Such correction can be applied using external information (if patient movement has been monitored during the scan), or relying on the images alone (coregistration, section 2.10).

1.2 Alzheimer's Disease

"Dementia" is the progressive decline in cognitive function beyond what might be expected from normal aging [44]. Its most common form is Alzheimer's Disease (**AD**). Dementia, and specifically AD, may be among the most costly diseases for society in developed countries [44]. It has therefore become a major topic of research.

AD is a neurodegenerative disease. Its onset increases dramatically with age. This may lead to the notion that its prevalence in the civilized world in recent decades is a side effect of prolonged life expectancy caused by better medical services. However, it could be shown that there are substantial differences between normally aged human brains and brains of subjects with AD [13].

At the time of this writing, the question of what is the primary cause of AD is yet undecided. There are a few competing theories, focusing on such different issues as deposition of amyloid plaques, tau protein abnormalities, late myelinisation [4]. The oldest theory proposes that AD is caused by reduced synthesis of the neurotransmitter acetylcholine [44].

[4] "Activity" here means radioactivity, not to be confounded with enzyme activity as of section 1.5

1.3 Cholinergic System and AChE

Acetylcholine (ACh) was the first neurotransmitter to be identified[5]. "Cholinergic" neurons make and release the transmitter at their axon terminals, while "cholinoceptive" neurons express ACh receptors on their postsynaptic membranes.

Cholinergic transmission plays a major role both in the central and peripheric nervous system of humans. It is established beyond doubt [16] that ACh concentrations in AD brains are well below ACh concentrations in age-matched healthy brains. This is explained with degeneration and death of cholinergic neurons. As a contribution to testing the different theories on AD, imaging techniques that monitor ACh in the living brain are of great interest. This is where PET comes in.

There is no PET tracer available for direct measurement of ACh. The "closest" related target would be the enzyme that makes ACh: Cholin Acetyl Transferase (ChAT). Yet there is no tracer for this enzyme, either. Another enzyme required for proper functioning of cholinergic synapses is Acetylcholine Esterase (**AChE**). Its function is to hydrolyze ACh in the synaptic cleft, setting free the AChE receptors and ending the transmission cycle. How essential this enzyme is for the synaptic function, is drastically demonstrated by the deadly chemical warfare agent sarin, it works by blocking AChE. As an ingredient of the cholinergic synapse, AChE can be used as a marker for cholinergic innervation of brain regions.

1.4 MP4A

AChE catalyzes hydrolysis of the ester bond between acetic acid and choline. Substrate analogues of ACh are therefore likely candidates for AChE tracers. The aim was an "irreversible tracer", one that stays trapped in the tissue after interaction with AChE. It must be able to cross the Blood Brain Barrier in order to reach the synaptic site; it needs an ester bond to be hydrolyzed by the enzyme, and the products (at least the one with the nuclide) should be hydrophilic to keep them from diffusing back to the blood stream. In addition, the tracer must be highly specific for AChE, excluding other causes for its hydrolysis.

Two tracers were designed to meet this specification:
1-[^{11}C]methyl-4- piperidyl acetate (**MP4A**) and 1-[^{11}C]methyl-4-piperidyl propionate (**MP4P**). Specifity of MP4A is higher (99%). Its rate of hydrolysis by AChE is high, making it suitable for brain regions of low enzyme activity. For medium and high activity regions, MP4P is better suited as its hydrolysis rate by AChE is lower, but its specifity for the enzyme is reduced as well.

[5] Sections 1.3 and 1.4 are based on [16, 19, 20]

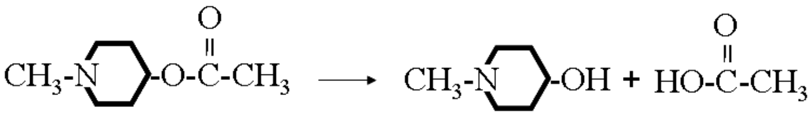


Figure 1.1: *The MP4A tracer and its hydrolysis, leading to N-methylpiperidinol and acetate. The marked ^{11}C is on the left side. Figure adapted from K. Herholz*

1.5 Kinetic Modelling

From dynamic PET data, TACs (section 1.1.5) can be extracted at the voxel or regional level. They provide information on nuclide accumulation, storage and washout in different parts of the brain. Kinetic models such as in **Figure 1.2** are used for quantitative analysis. The nuclide is seen as traveling between a small set of compartments, representing blood, tissue or different biochemical states. Travelling speed is controlled by kinetic constants (k_1, k_2, k_3 in the figure). Using the model, the constants can be estimated from the shape of the TACs. The quantity of interest is somehow

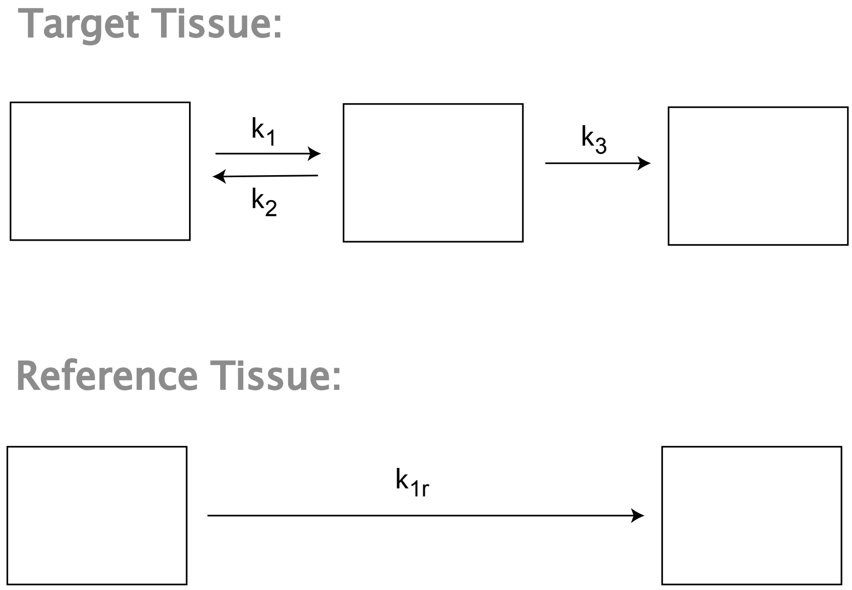


Figure 1.2: *Structure of the reference based irreversible tracer model. In the reversible model, k_3 is matched by a backward arrow with constant k_4. The reference part of the model arises from copying the target tissue part, renaming the constants k_{1r}, k_{2r}, k_{3r}. They are set to fixed "known" values. Assuming $k_{3r} = \infty$ leads to the simplification shown.*

linked to the constants: in the case of MP4A, k_3 is proportional to activity of AChE.

Compartmental models are in use with dynamic PET for various quantifications, such as blood flow, cerebral metabolic rate of glucose or neuroreceptor ligand binding. They all require information on tracer delivery via the blood stream. This so called “Blood Input Function” can be obtained independently by arterial blood sampling.

Later developments have produced a series of “reference tissue models” which avoid the need for blood sampling. They rely on comparing two Time Activity Curves: of the target region whose model parameters are being measured and the reference region whose parameters are known. For MP4A, the “reference tissue based irreversible tracer model” of Figure 1.2 is established. The details will be given in Figure 4.3 and chapter 3.

1.6 Voxel based and Region based kinetic Analysis

In MP4A PET studies, regional averages of the kinetic constant k_3 are compared between diagnostic subgroups of a patient sample in order to draw conclusions. A fundamental decision is between voxel- and region based procedures, where kinetic analysis is applied before or after regional averaging, respectively. Mathematically, the two operations do not commute, given the nonlinear nature of the model. When averaging is performed at the level of TACs, it is likely to produce a non-model-compliant curve, whose kinetic analysis leads to a biased result. The voxel-based approach is considered more ambitious in that it avoids this problem. Kinetic analysis takes place at the voxel level, leading to a so-called **parametric image** of the quantity of interest (k_3 in our case). Regional results are then obtained by averaging over the voxels of this image. This approach faces two difficulties: (1) given the large number of voxels in the brain, only fast methods of kinetic analysis are applicable, (2) TACs are noisier at the voxel level, creating additional problems for kinetic analysis.

(2) is often met by inserting a smoothing step, usually Gauss filtering, before kinetic analysis. While this reduces the level of noise, it partially sacrifices the advantages of the voxel based approach: (a) resolution is lost in parametric images, (b) again signal of different voxels gets mixed, albeit at closer proximity, and (c) it invites Partial Volume Effects (section 1.1.4.2). This even occurs between neighboring regions, presenting a disadvantage of the voxel based approach.

1.7 Aims of this Work

The author’s involvement in MP4A kinetic analysis started when a programmer was needed for implementing the voxel based COLOGNE method (chapter 3) as a feature of the VINCI package (section 2.3). He inherited an algorithm with a set of resources and policies from his predecessor in this work, G. Zuendorf [46], who had implemented it based on SPM99 [35] and MATLAB [28] software packages.

1.7.1 Extending the Application Range

The previous method employed a mask (the **Zuendorf Mask**) to remove 54.1 percent of the brain whose evaluation was considered problematic. The author was asked to make larger parts of the brain accessible, including the Hippocampus region whose role in the etiology of AD is of particular interest.

1.7.2 Redefining Image Preprocessing

Kinetic analysis is just one part of the workflow leading from dynamic PET scans to parametric images (see section 1.8). Porting the entire workflow required migrating all SPM99 components into the VINCI framework, which underwent a platform change at the same time. This had us caught up in technicalities for quite a while. In addition, the Zuendorf Mask had been an integral part of the image preprocessing stage. Removing it opened a Pandora's box of new questions, requiring definition and validation of preprocessing techniques and policies.

1.7.3 Making NLS available for Voxel based Analysis

Kinetic constants are determined by fitting a nonlinear model to the measured data. In mathematical terms, it comes down to multiple rounds of solving a nonlinear least squares (NLS)-problem. At the time of this work, there was widespread agreement that NLS is too expensive for voxel based analysis, the number of voxels per image ranging between hundreds of thousands and millions, depending on the image format. In addition, NLS solvers face the problem of suboptimal local attractors that might prevent finding the global optimum. Finally, a suitable starting point, or a method for generating one, must be found.

In response to these difficulties, a number of alternative algorithms, including aforementioned COLOGNE method, have become established (for more examples, see section 15.1). Since they do not meet the NLS specification, their behavior for noisy input is more or less undefined. Once we had discontinued using the Zuendorf Mask, areas of the brain were revealed where COLOGNE showed inconsistent behavior (section 12.1). This prompted the wish for an independent algorithm that could be used for validation, preferably one whose properties in the presence of noise have a clear mathematical definition.

1.7.4 Refining the Reference based irreversible Tracer Model

This model depends on existence of a brain region with idealistic properties (setion 3.2.2). Putamen is a small region thought to almost meet these requirements, Cerebellum is a large region that meets them to a lesser degree. It seemed therefore desirable to refine the model in such a way as to be able to handle nonideal reference regions, and use Cerebellum in the place of Putamen where necessary or useful.

1.7.5 Optimizing Procedures

As it is clear from sections 1.1.4 and 1.1.4.1, kinetic analysis of dynamic PET data is an uphill battle against the blurring effects of noise. Not only spatial resolution will suffer, but also precision of k_3. It is therefore paramount to optimize procedures of both the preprocessing stage and kinetic analysis with respect to noise propagation and stability. This requires the ability to measure success of ones efforts, i.e. precision of k_3.

1.7.6 Assessing Precision of k_3

Error estimation is mostly neglected in medical research. Complexity of the measured systems often leaves no other choice than viewing the method as a black box and relying on statistics for conclusions. The black box will be judged by (1) the reputation of components that went into its design and (2) statistical significance of the output, and how it relates to established knowledge. But, given the fact that sample sizes are small in MP4A PET studies, the author found this a bit unsatisfying.

While it remains impossible to factor in all circumstances that might lead to faulty results (section 1.8), it is within our reach to model the noise of a PET image and study its propagation through kinetic analysis, aiming at voxel- and region-based error estimates. A clearcut specification defining the "correct" result for noisy input, such as NLS, would be extremely helpful, creating another motivation for wanting an NLS solver.

1.7.7 An Outline

Chapter 2 is a collection of material for reference in later chapters. **Chapter 3** presents the established kinetic model for MP4A and MP4P tracers. It then proceeds with the COLOGNE method. New concepts are k_{3r}-correction and some implementation details. **Chapter 4** presents a fast implementation of a voxel based NLS solver for MP4A kinetic analysis. The chapter goes on to define a number of concepts that will assist in error assessment. **Chapter 5** deals with questions of starting points, break-off parameters and convergence properties. In **Chapters 6 and 7** both NLS and COLOGNE methods are validated using synthetic data: noiseless in chapter 6 to measure bias in isolation, and with simulated noise in chapter 7. **Chapter 8** introduces a voxel based error estimate and validates it against the simulator. **Chapters 9 and 10** cover the image preprocessing stages of both implementations, presenting the traditional workflow and the changes applied by the author. Chapters 11 to 14 continue the validation process, now using real PET data instead of simulations. Unexpected problems involving the reference curves motivated **Chapter 11**, while **Chapter 12** points out and handles a shortcoming of the model. **Chapter 13** shows some images and regional results. **Chapter 14** is devoted to optimizing the reference masks in response to aforementioned problem, and provides error estimates at the regional level.

1.8 Some general Remarks on Validation

The process leading to parametric k_3 images spans at least 5 steps of data generation and transformation (see Figure 1.3). Step 6 caters from there to regional results. At each of the steps there are various possibilities for error: assumptions, equipment or methods that fail to work out as expected.
By **Step 1**, we understand all it takes to produce the radioactive signal. It consists of a rain of gamma photons emanating from the patient's body, and its distribution in space and time contains the information we exploit. **Step 2** transforms this pattern to a set of *reconstructed images*, purportedly quantifying the distribution of radioactivity in the subject's head. **Step 3** is a sequence of image transformations, resulting in an array of *normalized frames* (section 9.1) that are congruent with each other. **Step 4** is the preprocessing stage, it leads to *"masked filtered frames"* from which voxel based TACs can be sampled. **Step 5** computes a "parametric k_3 image" from the TACs. **Step 6** extracts regional mean values, based on an atlas of brain regions.
"How do we know we are measuring AChE activity" is the key question. The answer relies on research in various disciplines, this work being just one contribution. Errors may creep in at any of the stages: **Step 1** relies heavily on the radiochemical purity of the tracer. The signal also depends on the tracer's biological properties, that have been studied by a number of researchers [19, 20, 21], leading to four assumptions spelled out in Chapter 3 and section 3.2.2. If any of these assumptions is not completely met, it will translate to false conclusions. **Step 2** is based on the scanning equipment and its reconstruction software. Both are extensively complex and rely on their own sets of assumptions, regarding detector properties, dead time of electronic components, assessment of attenuation, scatter and random fraction of the measured signal. We trust the manufacturer for proper handling of these factors. For **Step 3**, we rely on other software, which should be capable to properly align different images of the human

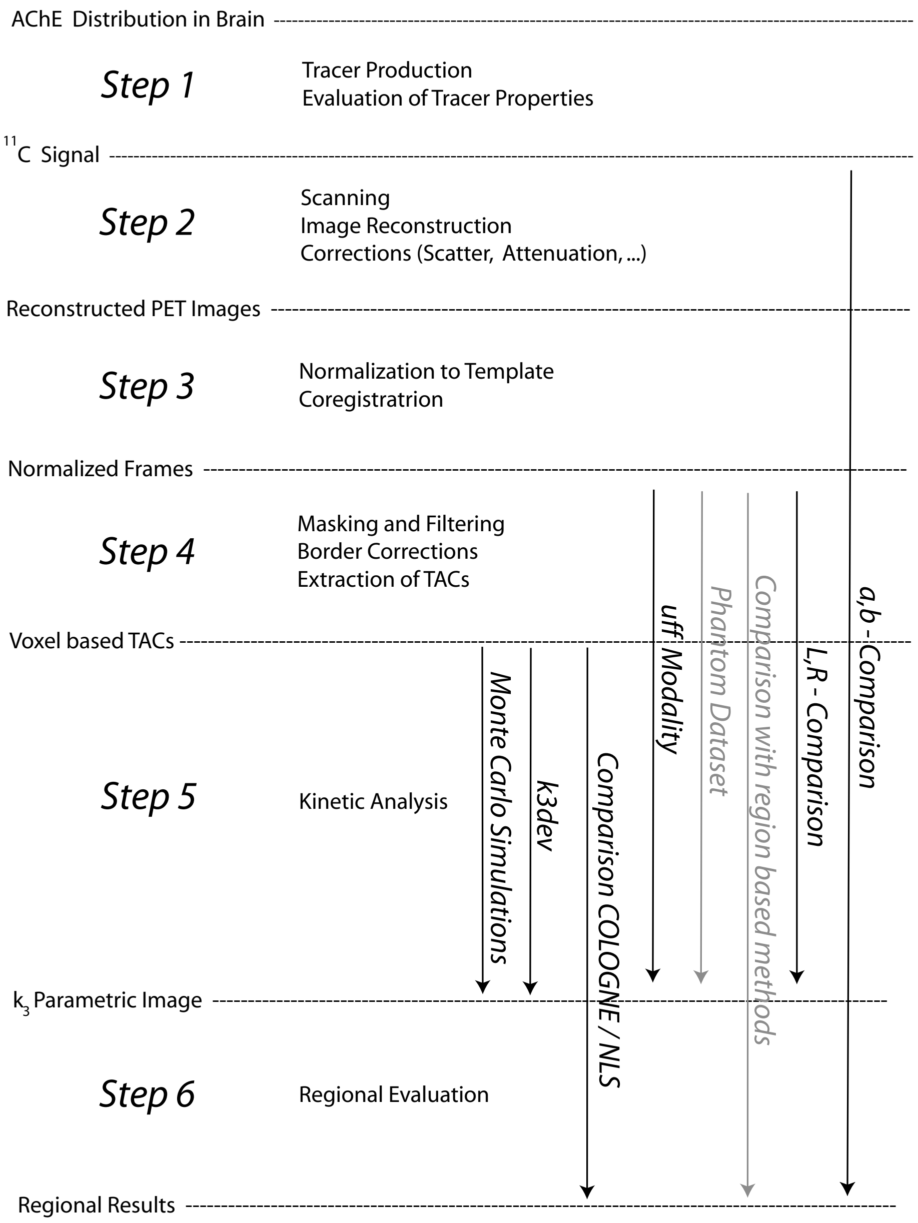


Figure 1.3: *Stages and steps of information processing, leading from the quantity of interest to regional data tables. Vertical arrows indicate possible validation efforts, some of which (in black) are part of this work.*

brain with each other (called *coregistration*), based on no other information than that of the images. In the context of VINCI, this work has been provided by [7, 8]. **Steps 4 and 5** are the subject of this work. The validity of **Step 6** is a complex issue, since it depends on a "Normalization Template", a brain atlas, a normalization algorithm (see again [7, 8]) and the assumption of constance of human brain anatomy: we expect all subjects to have their cerebral regions at the same relative location. Unfortunately, this is not fully the case. Anatomic differences between the patients are probably the single most significant source of bad regional results. In response to these difficulties, improvements in normalization procedures as well as templates and atlases are an ongoing field of research.

Another source of error that we ignore are Partial Volume Effects (section 1.1.4.2) resulting from the limits of scanner resolution. Methods taking this into account are called **Partial Volume Correction** (PVC). There exists theoretical and experimental work on PVC [2], but its application is demanding and most clinical studies run out of time before they can even consider it.

Chapter 2

Preliminaries

This chapter prepares some material for easy reference in later chapters. Much of it are well known facts in the PET community.

2.1 Decays and Activity

Becquerel [Bq] is a unit of radioactivity, it means decays per second. How much a Becquerel is in terms of atoms of a nuclide depends on its half life τ. It is computed by the formula

$$\frac{atoms}{Bq} = \frac{\tau}{1second \cdot ln(2)} \tag{2.1}$$

For ^{11}C with its half life of 1223 seconds, this makes 1764.4 atoms per Becquerel.

2.2 Voxels and Volumes

Tomographic images are stored as **"volumes"**, 3-dimensional arrays of pixels called **"voxels"**, each representing a cuboid sized a few millimeters across. Each voxel stores an "**intensity**" value coded in a 2 or 4 bytes integer or float format. In raw PET images, intensities correspond to units of specific activity (such as Becquerel per ml). Different file formats are in use, proprietary ones like **ECAT7** which is generated by Siemens tomographs, and other ones with differing amounts of header information like DICOM, Nifty, **Interfile**, Analyze. A **ROI** ("region of interest") is a subset of a volume, sometimes corresponding to an anatomical region of the brain. Binary **masks** are images of zeroes and ones, they are often used to identify ROIs.

2.3 VINCI and SPM

VINCI ("**V**olume **I**maging in **N**eurological Research, **C**o-Registration and ROIs **i**ncluded") was designed for the visualization and analysis of volume data generated by medical tomographical systems with special emphasis on PET [39, 41, 8]. Its development started at MPIFNF in 1999. It is both an image viewer capable of handling the above file formats, and an extensible application framework. Over the years it has accumulated functionality in many areas of image analysis, including coregistration and normalization (section 2.10). Newer versions depend on Trolltech's Qt library (now owned by Nokia Inc.) [32], allowing to reap the benefits of multi-platform development. MP4A kinetic analysis as presented in this work has been implemented in the C++ language as a VINCI plugin (see section 9). It is now part of the main distribution and available for the Linux, Solaris 10, MacOS X and MS Windows platforms.

SPM [35] is another well established software package developed and maintained by University College London, it is based on MATLAB [28] and offers similar functionality as VINCI.

2.4 Statistic Formulae

Consider a sample $x_1, \ldots, x_N$ of N values. The following concepts and abbreviations are used in this work:

- Mean value: $\mu = \frac{1}{N} \cdot \sum\limits_{i=1}^{N} x_i$
- Variance: $\sigma^2 = \frac{\sum\limits_{i=1}^{N} (x_i - \mu)^2}{N}$
- Sample Variance:

$$\frac{\sum\limits_{i=1}^{N} (x_i - \mu)^2}{N - 1} \tag{2.2}$$

- Standard deviation: $SD = \sigma = \sqrt{\sigma^2}$
- Coefficient of Variation, relative error: $COV = \frac{SD}{\mu}$

If all data are scaled with some factor c, then μ and SD will also scale with c, while σ^2 scales with c^2 and COV remains unchanged.

If not stated otherwise, the term "variance" refers to σ^2 and not to the sample variance. The latter is an estimator for the variance of a population whose expectancy is unknown, and from which we have a (small) sample.

P-values are indicators produced by various statistical tests. They give the probability that a specific feature of a sample could have occurred by chance. The feature is called *significant* if its P-value is less than 0.05, *highly significant* if it is less than 0.01.

2.5 Proportional Noise

is often used for modeling the noise of PET images. Suppose we are watching a quantity of some radioactive nuclide over a timespan where n atoms should decay. Then the number of atoms that will decay is Poisson distributed with mean value n. The variance of this distribution happens to be also n. More generally, we call it **Proportional Noise** whenever the variance is proportional to the signal n:

$$\sigma^2 \sim n$$

Hence this type of noise is characterized by $SD \sim \sqrt{n}$ and therefore

$$COV \sim \frac{1}{\sqrt{n}}$$

such that the brighter voxels or areas of an image have less relative error. The frames of a dynamic PET scan have Proportional Noise *before* being scaled towards units of specific activity by the reconstruction software.

2.6 Decay Correction

Chapter 3 presents the mathematical model for transport and degradation of the MP4A tracer. As a purely physiological model, it does not consider radioactive decay of the nuclide. Instead, it attempts to describe the behavior of *nonradioactive* ("cold") tracer. Yet in practice we need to use "hot" tracer. The fraction of the initially injected nuclide that still exists at time t is given by the **"Decay Function"**:

$$2^{-\frac{t}{\tau}} \tag{2.3}$$

where τ is the isotope half life. We assume that the remaining tracer is distributed in the body just like an equal dose of cold tracer would be distributed, had we injected it. Hence what the scanner finds at time t can be upscaled to its cold tracer equivalent by multiplication with $2^{\frac{t}{\tau}}$. This is called **Decay Correction**, and usually performed by the reconstruction software. The result is expressed in radioactive units, but remember it refers to nuclide at scan start time, not to the fraction that still exists at time t.

2.6.1 Implementation in clinical PET Scanners

The most consistent way of doing Decay Correction is to take every count times $2^{\frac{t}{\tau}}$. For instance: if a count is registered after 3 half lifes of the isotope, it is worth 8 counts, 7 of which represent atoms that ought to show up in the same area, but have decayed in the meantime.

However, this approach requires either real-time computations, or recording the time information with every count. For this or other reasons, it has not been adopted in clinical PET scanner software. Instead, the scanners operate in "histogram mode" where the counts are simply added up, discarding time information. A global correction is then applied to every frame. The best correction factor available after such loss of information is

$$\frac{t_2 - t_1}{\int\limits_{t_1}^{t_2} 2^{-\frac{t}{\tau}} dt} \tag{2.4}$$

where t_1,t_2 are the start and end times of the frame. This is the integral of the constant function 1, divided by the integral of the decay curve, over the time interval of the frame. After solving the integral, this **Decay Correction Factor** becomes

$$\frac{ln(2) \cdot (t_2 - t_1)}{\tau \left(2^{-\frac{t_1}{\tau}} - 2^{-\frac{t_2}{\tau}} \right)} \tag{2.5}$$

It is still close to 8 if t_1 and t_2 are close to 3 half lifes.

The measured counts are taken times this factor, and then undergo conversion toward units of specific activity, such as Bq/ml. Since the Becquerels are counts per second, this final conversion contains division by the frame length. So the overall factor applied by the reconstruction software is proportional to

$$\frac{1}{2^{-\frac{t_1}{\tau}} - 2^{-\frac{t_2}{\tau}}} \tag{2.6}$$

This will be called **"Combined Correction"** in this work, as it accounts for both decay and the differences in frame length. It converts from measured counts to concentrations of initially injected nuclide, which is the entity used by kinetic modelling.

2.7 Decay Weighting

According to the Decay Function (2.3), the proportion $2^{-\frac{t_1}{\tau}} - 2^{-\frac{t_2}{\tau}}$ of the initial dose decays between times t_1 and t_2. Let

$$w_i := 2^{-\frac{t_i}{\tau}} - 2^{-\frac{t_{i+1}}{\tau}} \tag{2.7}$$

where t_i and t_{i+1} are the start and end times of the i^{th} scan frame. This sequence is called **Decay Weighting** in this work. Note that the weights are inverse to the Combined Correction factor of equation (2.6).

2.7.1 Decay Proportional Noise

Proportional Noise as of section 2.5 is correct for modelling voxel- or region-based noise, and has been used for the simulator of chapter 7 and the error formula of chapter 8. But we also need an assessment for average noise of whole frames. Hence we make the additional simplification of assuming levels of specific acticity constant over time. For justification, consider the plots of section 14.1.13 where a plateau is reached and maintained after a few minutes.

Again we start by assuming Proportional Noise before Combined Correction is applied. The average signal of a frame is then proportional to the number of detected decays, which are in turn proportional to the Decay Weights of equation (2.7):

$$``Signal'' \sim w_i$$

Then, by equations of section 2.5, we have the proportionalities:

$$\sigma^2 \sim w_i$$

$$SD \sim \sqrt{w_i}$$

$$COV \sim \frac{1}{\sqrt{w_i}}$$

As the following sections show, this type of noise makes it sometimes favorable to apply Decay Weighting.

2.7.2 Adding up Frames

Suppose we want to compute a weighted sum image of several frames, and wish to minimize the relative error in every voxel of that image. For frames with Proportional Noise it can be proven rigorously (see appendix A.1) that the best summation method is unweighted.

Reconstructed dynamic PET data arise from frames with Proportional Noise by applying Combined Correction, which amounts to scaling with $\frac{1}{w_i}$ (i=1,...,N). Hence the best weighted summation is the one that undoes Combined Correction, by applying w_i (i=1,...,N) as summation weights - which is the Decay Weighting.

2.7.3 Weighted least Squares

Least Squares Approximations to an N-dimensional target vector minimize the sum of the components' squared differences. But if the components are affected by different levels of noise, it will do no good trying to approximate them all to the same precision. Instead, one would seek a closer approximation of the components containing the more precise measurements. A

weighting **u** is introduced for that purpose, so we are minimizing the penalty function $\sum\limits_{i=1}^{N} u_i \cdot d_i^2$, where d_i is the distance between target and approximation on the i^{th} component.
It seems reasonable to choose **u** such that the individual penalties

$$u_i \cdot d_i^2$$

become the same for one standard deviation on each frame. Hence we want the $u_i \cdot \sigma_i^2$ to be equal, which can be achieved by letting

$$u_i := \frac{1}{\sigma_i^2} \tag{2.8}$$

In the case of Decay Proportional Noise, where we have $\sigma_i^2 \sim w_i$ before Combined Correction and $\sigma_i^2 \sim \frac{1}{w_i}$ afterwards, this becomes $u_i \sim w_i$. Hence we recommend to use Decay Weighting for least squares approximations of PET frames as delivered by the reconstruction software.

2.7.4 Isotropy of Noise

Minimizing Decay Weighted squares is the same as minimizing distances in a space whose metric is defined by the inner product of equation (4.7). With Decay Proportional Noise, we have

$$SD \sim \sqrt{w_i}$$

for the signal before Combined Correction, and

$$SD \sim \frac{1}{\sqrt{w_i}}$$

afterwards. We therefore have standard deviations proportional to the unit vectors (equation (B.10)) of this metric. This implies equal Gaussian noise in all directions. Now the N-dimensional cartesian product of normal distributions with $\mu = 0$ and equal σ results in a function that depends only on the distance from the origin. This is a consequence of the identity

$$\prod_{i=1}^{N} exp(-x_i^2) = exp\left(-\sum_{i=1}^{N} x_i^2\right) \tag{2.9}$$

where $\sum\limits_{i=1}^{N} x_i^2$ is the square of the distance. So we have a noise cloud of perfect spherical symmetry. This motivated the design of $k3var_i$ in section B.1.

2.8 Similarity Measures

are used in this work for comparison of data vectors - huge ones consisting of the voxels of an image, or small ones consisting of measurements. They will be called ‘shift’, ‘dist’ and ‘corr’.

‘shift’ is the signed difference between the two mean values of either dataset. It indicates “bias”, a systematic drift in one direction.

‘dist’ is the Euclidean Distance scaled with $\frac{1}{\sqrt{N}}$:

$$`dist' := \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2} \tag{2.10}$$

It can be looked upon as a sort of average distance.

‘corr’ is the correlation coefficient of the cloud of points whose coordinates are the data of sets 1 and 2. It is indifferent to shifting or scaling of either data set. If applied to images, it compares their “content”, and it will diminish if either image is disfigured by noise.

Mutual Information is mentioned just for completeness, as it plays no role in this work. It is a similarity measure used for automatic inter-modality coregistration (section 2.10), it can compare images even if corresponding structures follow different patterns of brightness.

2.9 Gauss Filtering

is a smoothing technique often applied to PET images during or after reconstruction. It will reduce spatial noise of a single frame. This also leads to lower noise in the fourth dimension, the time, where it is of interest for MP4A studies: improving the quality of the Time Activity Curves from which k_3 is computed. Every voxel of the filtered image is replaced by a weighted mean of the original voxels in its neighborhood. This is a linear combination with positive coefficients, which add up to 1 and together are called the **filter kernel**. Gauss Filter kernels are shaped like a 3-dimensional normal distribution. The only free parameter is **FWHM**, “full width at half maximum”. The density of the normal distribution centered in 0 with standard deviation σ is given by

$$\varphi(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{x^2}{2\sigma^2}} \tag{2.11}$$

The abscissae of its half maximum are at $\pm\sqrt{2 \cdot ln(2)}\sigma$ (not to be confounded with its points of inflection which are at $\pm\sigma$). By definition, FWHM equals the distance between these abscissae, which is $\sqrt{8 \cdot ln(2)}\sigma \approx 2.3548 \cdot \sigma$.

So, for given FWHM, we obtain σ by division through 2.3548. Using this σ and formula (2.11), we compute the filter kernel: at the k^{th} voxel in x direction from the voxel of interest, we plug in kv for x, where v is the voxel size in x direction. Let the voxel be isotropic and i, j, k denote the offsets in x,y and z direction: then we have a distance of $v \cdot \sqrt{i^2 + j^2 + k^2}$. We want spherical symmetry, where the kernel depends exclusively on the distance from the voxel of interest. So it should be computed by plugging the distance into formula (2.11) and replacing the scalar factor by its third power, since we are in 3 dimensions. Thus, we get for the kernel at voxel (i,j,k)

$$\frac{1}{(\sigma \cdot \sqrt{2\pi})^3} \cdot e^{-\frac{v^2 \cdot (i^2+j^2+k^2)}{2\sigma^2}} \tag{2.12}$$

2.9.1 Decomposition into one dimensional Steps

Note that, as a consequence of the identity

$$e^{-v^2 i^2} \cdot e^{-v^2 j^2} \cdot e^{-v^2 k^2} = e^{-v^2 (i^2+j^2+k^2)}$$

this happens to be the same as the product $\varphi(iv) \cdot \varphi(jv) \cdot \varphi(kv)$ with φ taken from equation (2.11). We thus can replace direct application of a 3-dimensional kernel by breaking down the filtering process into three separate passes, during which the image is filtered in x, y and z direction. This brings considerable speedup over a single pass implementation. The speedup factor is the number of voxels of three 1-dimensional kernels, divided by the number of voxels of one 3-dimensional kernel. This starts to pay off for large FWHM, small voxel sizes or high cutoff parameters, and has therefore been used in the Gauss Filter implementations both of VINCI and SPM.

2.9.2 Scaling

The integral both of the one-dimensional and the 3-dimensional kernel (equations (2.11) and (2.12)) is exactly 1. The sum of kernel coefficients (which is a discretized version of this integral) will therefore be close to 1 to begin with. What difference there is, is compensated for by scaling.

2.9.3 Cutoff

Since we do not want to sample the whole original image for every voxel of the target image, cutoff is applied at a certain distance from the voxel of interest. The distance is usually set to a fixed number k of standard deviations. **Table 2.1** has been computed by integration of the 3-dimensional bell function, it shows what percentage of the total integral is under a cubic domain extending k times σ away from the voxel of interest in each dimension and direction. Thus, at only twice the computational cost of 95%, we obtain an almost perfect Gauss Filter and do not have to worry about cutoff artefacts.

k	percentage covered
2.11405	90.0
2.38774	95.0
2.93416	99.0
3.58783	99.9
4.1494	99.99
4.64913	99.999

Table 2.1: *Cutoff yield of cubic Gauss Filter domains*

Additional cutoff will take place at the image borders and corners. There we lack input from nearby voxels, so scaling is needed to avoid darkening in these places.

In the VINCI implementation, cutoff (using k=4.64913) and scaling are applied during each of the three 1-dimensional steps. Depending on their closeness to the border, the one dimensional kernels are used with appropriate scaling factors to have their coefficients add up to 1.

2.9.4 Filtering twice

For the bell curve φ of equation (2.11), it is known that the convolution with itself yields the same type of function. More precisely,

$$\int_{-\infty}^{\infty} \varphi_1(x)\varphi_2(t-x)dx = \varphi_3(t) \tag{2.13}$$

and $\sigma_3^2 = \sigma_1^2 + \sigma_2^2$ where the σ_i are the σ's of the φ_i.

When translating this finding to kernels and filtering, then convolution comes down to filtering twice, applying two kernels sequentially to one image. Since FWHM=2.3546 $\cdot\, \sigma$, the relation between the σ's holds for the FWHM as well. So filtering twice with kernel sizes of $FWHM_1$ and $FWHM_2$ is equivalent to filtering once with a kernel size of $FWHM_3$, which is the pythagoreian sum of $FWHM_1$ and $FWHM_2$:

$$FWHM_3 = \sqrt{FWHM_1^2 + FWHM_2^2} \tag{2.14}$$

By writing equation (2.12) in its decomposed form (see section 2.9.1) and observing that everything commutes, one can show that this result applies to 3-dimensional filtering as well.

2.9.5 An Alternative

Instead of evaluating the 3-dimensional bell curve at the center of each voxel like in formula (2.12), one could also use the integrals of the bell curve over the voxel domains as kernel coefficients. In fact this seems to be slightly more correct. This approach can also be decomposed into 1-dimensional steps and therefore allows fast implementation.

2.10 Coregistration and Normalization

are techniques for aligning images of the human brain with each other. One may have images of the same subject but of different **modalities** (CT or MR against PET, or PET images with different tracers), then inter-modality coregistration is needed. Inter-subject coregistration is often used in scientific studies, where a region should appear in the same place for all subjects. Usually, one of the images is called a **"template"**, to which the others are **"normalized"** in order to make them comparable.

An important difference between same-subject and inter-subject coregistration of brain scans is that, for the former, it suffices to consider translation and rotation, since the size and shape of the brain is the same. Images from clinical scanners usually come with valid metric information where distances, angles and volumes can be measured. But the head location will be "unknown" and different from image to image, as it depends on how the patient was positioned in the scanner. In order to realign two images of the same subject, all possible translations (3 degrees of freedom) and rotations (3 more) must be considered, this is called **Rigid Body Coregistration**.

In inter-subject coregistration there may also be differences in head size and shape. So 3 stretch factors are added to support size adjustment independently in 3 directions of space. If these directions are fixed to the coordinate axes, it requires 3 degrees of freedom, and 3 more if an arbitrary orthonormal system of stretch axes is allowed. We speak of **"Affine Normalization"** with 9 or 12 degrees of freedom. By 12 degrees of freedom, the set of affine mappings is exhausted, and anything beyond will be nonlinear.

2.10.1 Automatic Coregistration

Coregistration can be "manually" performed by comparing images in a viewer with support for affine operations. In many instances this is the safest method. However, for the human brain with its rich internal structure, there exist algorithms for automatic coregistration and normalization that can do as well as any human operator [7, 8]. They determine the set of transformation parameters that maximizes similarity (section 2.8) between image 2 and transformed image 1.

Users need to make the right choices: use Rigid Body Coregistration for intra-subject image alignment, Affine Normalization for inter-subject situations. Use Mutual Information as a similarity measure for inter-modality coregistration, while the Correlation Coefficient is good enough for same modalities. Automatic procedures will not work in combined inter-subject inter-modality situations. It may then be possible to get things running again by using intermediates as bridges, an example is in section 9.2.1.

2.10.2 Image Resampling

Once the optimum transformation has been found, image 1 may be "resampled" to create a version of itself (image 1') in the coordinate space of image 2. Such resampling can be coupled with a change of the volume format (the number and size of the voxels in the 3 coordinate directions). The procedure is implemented in VINCI as follows: start with an empty volume in the format of image 2, and iterate over its voxels. By the chosen transformation, map every voxel **v** to its corresponding location in image 1. Mostly, that location will not be the center of a voxel, hence the intensity value is computed by "threelinear interpolation" of the intensities of up to 8 voxels. The resulting mean value is assigned to **v**.
This method has two implications:

1. Resampling comes with a degree of smoothing and loss of image resolution by averaging. So one tries to keep the number of resamplings low.

2. Image intensities are not corrected for volume effects. That is to say, the intensity of a region stays the same even if the image changes in size. Therefore, in the resampled image, the regional readings of specific activity (Becquerel per ml) are the same as in the original image, but the total amount of radioactivity is misrepresented.

2.11 Patient Samples

A suspected precursor stage of AD is "mild cognitive impairment" (**MCI**), where patients complain of memory deficits while other cognitive functions are still intact. In 2007, a study involving 12 elderly volunteers diagnosed with MCI was conducted at MPIFNF, Cologne. Each subject received an MP4A PET scan and an MRI scan, and underwent a battery of neuropsychological tests. A similar study, involving 12 MCI subjects and 5 normal controls, was performed at the San Raffaele Hospital in Milan. The studies underwent comparative analysis under the umbrella of DiMI Workpackage 8.2. **DiMI**, "Diagnostic Molecular Imaging", is an EC funded initiative designed to coordinate research on various diseases involving multimodal imaging techniques.

In this work, subjects will be referenced by their DiMI identification codes. We define 4 patient samples for later reference (see below). They were all scanned using the "traditional" framing of section 6.1.1.1. For patients of Samples 1 and 2 we used an ECAT HR scanner (CTI/Siemens, Knoxville, Tennessee, USA), voxel size: 2.202 × 2.202 × 3.125 mm, volume size: 128 × 128 × 47 voxels. A GE Advance scanner was used in Milan for Sample 3 and 4 subjects, voxel size: 2.5 × 2.5 × 4.25 mm, volume size: 128 × 128 × 35 voxels. Reconstruction protocols for Sample 3 were different from Samples 1 and 2 in that they inferred more smoothing to the raw PET data.

- Cologne MCI subjects of 2007 (=**Sample 1**)
  DiMI ID codes:
  M01001, M01002, M01003, M01004, M01005, M01007, M01008, M01009, M01010, M01012, M01013, M01014.

- Cologne subjects of 2002 and 2003 (=**Sample 2**)
  Diagnostic groups and DiMI ID codes:

  - AD: M01051, M01053, M01054, M01056, M01057, M01058, M01059, M01061, M01062, M01063, M01064
  - Normal Controls: M01068, M01081
  - MCI: M01089, M01090, M01091, M01092

- Milan subjects:
  Diagnostic groups and DiMI ID codes:

  - MCI: M36002 through M36012 minus M36007 (=**Sample 3**)
  - Normal Controls: M36022 through M36025 (=**Sample 4**)

Chapter 3

Kinetic Model and COLOGNE Method

The MP4A kinetic model used in this work is based on 4 assumptions, beginning with

Assumption 1 (Ignoring Blood Volume)
The measured ^{11}C signal comes from tissue alone

Assumption 2 (Tracer Irreversibility)
Once the tracer is hydrolyzed, its ^{11}C can no longer cross the Blood Brain Barrier

Assumption 3 (Tracer Specificity)
The speed of hydrolysis is proportional to AChE enzyme activity

Assumption 1 is a simplification which is partly justified by the fact that (1) the average "blood volume"[1] in brain is only 4 to 5 percent [13] and (2) specific activity in blood plasma is known to decline rapidly after an initial burst following injection of the tracer. Therefore, contribution of signal from the blood to the total signal measured is presumed negligible by most researchers. We will pick up this issue in sections 12.1 and 15.10.1. Assumptions 2 and 3 are founded on studies on the properties of MP4A by Irie et al. [20, 19]. Assumption 2 provides justification for setting k_4 to zero, leading to the irreversible tracer model of Figure 1.2.

3.1 Deriving the Model Functions

A detailed view of the model is shown in **Figure 4.3**, whose 3 compartments refer to the following concentrations:

[1] the inner volume of the capillaries as a fraction of total tissue volume

1. Authentic tracer in blood plasma (C_{pl})
2. Authentic tracer in brain tissue (C_{ta})
3. Hydrolyzed tracer in brain tissue (C_{th})

The kinetic constants $\mathbf{k_1}$, $\mathbf{k_2}$, $\mathbf{k_3}$ give the speed of tracer diffusion from plasma to tissue, from tissue to plasma, and of tracer hydrolysis in tissue. They are in units of $\min^{-1}$. For instance, k_2=0.1 and k_3=0.2 means that back diffusion occurs at a rate of 10 percent per minute and hydrolysis at 20 percent per minute, both percentages referring to the concentration of authentic tracer in tissue. Matters are slightly more complicated for k_1, which unlike k_2 is not merely a permeability constant, but also depends on "perfusion" which is the rate of blood delivery to tissue. It is therefore more correctly expressed in units of $\frac{\text{ml of plasma}}{\text{min} \cdot \text{ml of tissue}}$. In practical calculations of this work, k_1 never occurs in absolute terms, but in dimensionless ratios of two such constants.

Total tracer in tissue equals $\mathbf{C_t} = \mathbf{C_{ta}} + \mathbf{C_{th}}$, and by Assumption 1, the signal is proportional to C_t. Hydrolyzed tracer in blood plasma need not be considered because of Assumptions 1 and 2. $\mathbf{C_{pl}}$ is the **(Blood) Input Function**, it is known by direct or indirect measurement. C_{ta} grows only by tracer crossing the Blood Brain Barrier, at a rate proportional both to C_{pl} and k_1. C_{ta} decreases by tracer crossing back into the plasma, and tracer being hydrolyzed. So the rate of decrease is $(k_2 + k_3)C_{ta}$. $k_3 C_{ta}$ is also the speed at which C_{th} increases. Because of Assumption 2, there is nothing to make it decrease.
So we have the following system of differential equations for C_{ta} and C_{th}:

$$C'_{ta} = -(k_2 + k_3) \cdot C_{ta} + k_1 C_{pl} \tag{3.1}$$

$$C'_{th} = k_3 C_{ta}$$

The relevant solution is given in equation (3.4). For the sake of completeness, we include its derivation: let $\boldsymbol{k}$ be a shorthand for $k_2 + k_3$. The first equation does not contain C_{th} and can therefore be solved in isolation. Without the inhomogeneity $k_1 C_{pl}$, the solution is

$$y_{hom}(t) = r \cdot e^{-kt}$$

where r is a constant. To find a single solution of the inhomogenous problem, we replace r by a function $r(t)$ and try:

$$y_{inhom}(t) = r(t) \cdot e^{-kt}$$

with its derivative

$$y'_{inhom}(t) = (r'(t) - kr(t)) \cdot e^{-kt}$$

Substituting this for C_{ta} and C'_{ta} in equation (3.1) leads to

$$r'(t) = k_1 \cdot C_{pl}(t) \cdot e^{kt}$$

and r is an integral function of the right side. Since there is no tracer expected in the tissue at time 0, we choose r(0)=0 and obtain

$$r(t) = k_1 \cdot \int\limits_0^t C_{pl}(s) \cdot e^{ks} ds$$

$$C_{ta}(t) = r(t)e^{-kt} = k_1 e^{-kt} \cdot \int\limits_0^t C_{pl}(s) \cdot e^{ks} ds \tag{3.2}$$

Then, by the second line of (3.1)

$$C'_{th} = k_1 k_3 \cdot e^{-kt} \cdot \int\limits_0^t C_{pl}(s) \cdot e^{ks} ds$$

C_{th} is an integral function of C'_{th} with respect to t:

$$C_{th} = k_1 k_3 \int \underbrace{e^{-kt}}_{\uparrow} \cdot \underbrace{\int\limits_0^t C_{pl}(s) \cdot e^{ks} ds}_{\downarrow} \, dt$$

Partial integration as indicated by the arrows leads to

$$C_{th} = k_1 k_3 \cdot \left(-\frac{1}{k} \cdot e^{-kt} \cdot \int\limits_0^t C_{pl}(s) \cdot e^{ks} ds \right) - k_1 k_3 \cdot \left(\int -\frac{1}{k} \cdot e^{-kt} C_{pl}(t) \cdot e^{kt} dt \right)$$

where the right brace is an unspecified integral function. At time 0, there ought to be no hydrolyzed tracer, so we can set the integration limits to 0 and t, obtaining:

$$C_{th}(t) = -\frac{k_1 k_3}{k} \cdot e^{-kt} \int\limits_0^t C_{pl}(s) e^{ks} ds + \frac{k_1 k_3}{k} \int\limits_0^t C_{pl}(s) ds \tag{3.3}$$

By substituting (3.2) and (3.3) into $C_t = C_{ta} + C_{th}$, we have

$$C_t(t) = \left(k_1 - \frac{k_1 k_3}{k} \right) e^{-kt} \cdot \int\limits_0^t C_{pl}(s) e^{ks} ds + \frac{k_1 k_3}{k} \int\limits_0^t C_{pl}(s) ds$$

where the brace equals $\frac{k_1 k_2}{k}$ because $k = k_2 + k_3$.
e^{-kt} can be drawn into the first integral and united with e^{ks}. So the final result is

$$\boxed{C_t(t) = \frac{k_1 k_2}{k} \cdot \int\limits_0^t C_{pl}(s) e^{k(s-t)} ds + \frac{k_1 k_3}{k} \cdot \int\limits_0^t C_{pl}(s) ds} \tag{3.4}$$

3.2 Obtaining the Input Function

C_t depends on the Input Function C_{pl} and the kinetic constants k_1, k_2, k_3. Conversely, if C_t and C_{pl} are known, k_1, k_2, k_3 can be computed by the methods of this and the following chapter. C_{pl}, the concentration of unhydrolyzed tracer in the plasma, is required as a global input to these procedures.

3.2.1 Blood Sampling

Until recently [15], C_{pl} was obtained directly by arterial blood sampling. But the tracer is hydrolyzed in the plasma as well, so it did not suffice to count radioactivity. The tracer had to be chemically separated from its degradation products, which was neither convenient nor precise. In addition, drawing arterial blood during PET acquisition may lead to patient discomfort and movement.

3.2.2 Reference Regions

For these reasons, new ways for obtaining C_{pl} were sought, and in [15] a method based on a reference region was published. Such a region is expected to satisfy

Assumption 4 (Ideal Reference Region)
There is no backflow of radioactivity from the reference region to the plasma

In the model, this will happen if either k_2 is zero or k_3 is infinity, where every tracer molecule is hydrolyzed immediately after crossing the Blood Brain Barrier. Substituting either of the two in equation (3.4), leads to

$$C_r(t) = k_{1r} C_i(t) \tag{3.5}$$

where C_r is the C_t of the reference region, k_{1r} is its local k_1 and

$$C_i(t) := \int\limits_0^t C_{pl}(s) ds \tag{3.6}$$

is a special integral function of C_{pl}. Then by measuring C_r, we get to know C_i up to a constant, and thus we know its derivative C_{pl}.

For Cerebellum and Putamen, k_3 values of 0.65 and 4, respectively, have been reported in literature (section 13.1). Their k_2 is assumed close to 0.1. That makes them suitable, although not ideal, as reference regions. However, if k_2 and k_3 of the reference region are known, there is a way to reconstruct an ideal reference curve from the measured data:

3.2.3 k_{3r}-Correction

Let C_r be the reference curve of an ideal reference region for which equation (3.5) applies, and C_{rm} a real reference curve measured from Cerebellum or Putamen. It is connected with C_{pl} by equation (3.4), written for the kinetic constants k_{1r}, k_{2r}, k_{3r} and k_r=k_{2r}+k_{3r} of the reference region:

$$C_{rm}(t) = \frac{k_{1r}k_{2r}}{k_r} \cdot \int\limits_0^t C_{pl}(s)e^{k_r(s-t)}ds + \frac{k_{1r}k_{3r}}{k_r} \cdot \int\limits_0^t C_{pl}(s)ds \tag{3.7}$$

The question is, how can C_r be computed from C_{rm} and k_{2r}, k_{3r}. For abbreviation, let

$$q_r := \frac{k_{3r}}{k_r}$$

leading to

$$1 - q_r = \frac{k_{2r}}{k_r}$$

Using q_r and C_i, equation (3.7) assumes the following form:

$$C_{rm}(t) = k_{1r}(1-q_r) \cdot \int\limits_0^t \underbrace{C_{pl}(s)}_{\uparrow}\underbrace{e^{k_r(s-t)}}_{\downarrow}ds + k_{1r}q_r \cdot C_i(t) \tag{3.8}$$

We expand by partial integration, as the arrows indicate:

$$C_{rm}(t) = k_{1r}(1-q_r)\left[C_i(s)e^{k_r(s-t)}\right]_{s=0}^{s=t} - k_{1r}(1-q_r)\int\limits_0^t C_i(s)k_re^{k_r(s-t)}ds + k_{1r}q_rC_i(t)$$

By plugging in s=0, s=t to the first part and observing $C_i(0) = 0$, this part becomes $k_{1r}(1-q_r)C_i(t)$ and can be united with the last summand to give $k_{1r}C_i(t)$. Hence we have

$$C_{rm}(t) = k_{1r}C_i(t) - k_{1r}k_r(1-q_r)\int\limits_0^t C_i(s)e^{k_r(s-t)}ds$$

Now by definition of q_r,

$$k_r(1-q_r) = k_{2r}$$

Using this and commuting the equation sides, we arrive at

$$k_{1r}C_i(t) = C_{rm}(t) + k_{2r}\int\limits_0^t k_{1r}C_i(s)\cdot e^{k_r(s-t)}ds$$

$$C_r(t) = C_{rm}(t) + k_{2r}\int\limits_0^t C_r(s)\cdot e^{(k_{2r}+k_{3r})(s-t)}ds \tag{3.9}$$

C_r is now seen in two places: on the left hand side, and inside the integral. This allows the following iterative method to be used. Begin with $C_r = C_{rm}$. Plug this into the right side and evaluate the integral, using the true values of k_{2r} and k_{3r}. Equation (3.9) returns an update for C_r, the second approximation. The procedure is repeated until convergence is reached. For computational details, see section 3.4.1.1.

3.3 The COLOGNE Method

Given a Blood Input Function C_{pl} and a local Time Activity Curve C_t, k_1/k_{1r}, k_2 and k_3 can be obtained by fitting the right hand side of equation (3.4) to C_t. Since a least squares fit was deemed too expensive for voxel based computation with commercial software packages, a number of faster methods has been published by different authors [5, 29, 36, 30]. One of them, developed by Herholz and Zuendorf in Cologne, will be called the COLOGNE method. Our presentation follows [46] with notational modifications. In analogy to section 3.2.3, define

$$q := \frac{k_3}{k} \tag{3.10}$$

leading to

$$1-q = \frac{k_2}{k} \tag{3.11}$$

and rewrite equation (3.4) using q and C_i:

$$C_t(t) = k_1(1-q)\cdot\int\limits_0^t C_{pl}(s)e^{k(s-t)}ds + k_1qC_i(t) \tag{3.12}$$

Now consider its quotient with $C_r(t)$=$k_{1r}C_i(t)$:

$$\frac{C_t(t)}{C_r(t)} = \frac{k_1(1-q)\int\limits_0^t C_{pl}(s)e^{k(s-t)}ds}{k_{1r}C_i(t)} + \frac{k_1qC_i(t)}{k_{1r}C_i(t)}$$

$$= \frac{k_1}{k_{1r}}(1-q) \cdot \frac{\int\limits_0^t C_{pl}(s) e^{k(s-t)} ds}{C_i(t)} + \frac{k_1}{k_{1r}} q$$

So we have

$$\frac{C_t(t)}{C_r(t)} = A \cdot w_k(t) + B \tag{3.13}$$

where

$$w_k(t) := \frac{\int\limits_0^t C_{pl}(s) \cdot e^{k(s-t)} ds}{C_i(t)} \tag{3.14}$$

$$A := \frac{k_1}{k_{1r}}(1-q), \quad B := \frac{k_1}{k_{1r}} q \tag{3.15}$$

The left hand side of equation (3.13) is known from the image data. Given an estimate for k, $w_k(t)$ can be computed, too. First we transform it by partial integration such as to replace C_{pl} by C_i:

$$w_k(t) = \frac{\int\limits_0^t C_{pl}(s) \cdot e^{k(s-t)} ds}{C_i(t)}$$

$$= \frac{\left[C_i(s) e^{k(s-t)}\right]_{s=0}^{s=t}}{C_i(t)} - \frac{\int\limits_0^t C_i(s) \cdot k \cdot e^{k(s-t)} ds}{C_i(t)}$$

The first fraction equals 1 since $C_i(0) = 0$. Expanding the second fraction with k_{1r}, we get

$$w_k(t) = 1 - k \cdot \frac{\int\limits_0^t C_r(s) \cdot e^{k(s-t)} ds}{C_r(t)} \tag{3.16}$$

From C_r, which is known as a value table for discrete times, the enumerator is computed by interpolation and integration in ways we discuss in 3.4.1. So (3.16) allows to compute a value table for w_k. If k was guessed truthfully, (3.13) ensures it will correlate perfectly with the value table for $\frac{C_t}{C_r}$.

Hence, we determine k such that it maximizes the correlation between the two sides of (3.13). For the optimal k, A and B are then determined by linear regression. Since by (3.15), $\frac{k_1}{k_{1r}} = A + B$, we can compute this first and then $q = \frac{k_{1r}}{k_1} B$. By (3.10), we have $k_3 = qk$, which gives us k_3. Finally, $k_2 = k - k_3$.

3.4 Implementation Details

3.4.1 Interpolation and Integration

For the COLOGNE method,

$$\int_0^t C_r(s) \cdot e^{k(s-t)} ds \tag{3.17}$$

of equation (3.16) must be evaluated. C_r is given as a value table for discrete times t=$t_1, t_2, \ldots, t_N$. For each t_i in the place of t, (3.17) is computed and used in (3.16) to obtain w_k as a value table. To make integration possible, C_r must be interpolated on the continuum. Linear interpolation would systematically underestimate the integral, because C_r is concave. We therefore interpolate by fitting a natural cubic spline through the set of points

$$\{(t_i|C_r(t_i)) \mid i = 1, \ldots, N\} \tag{3.18}$$

By definition, that is a C^2-continuous 3^{rd} order spline changing formula in $t_1, \ldots, t_N$ and degenerating to a 1^{st} order polynomial left of t_1 and right of t_N. Natural cubic splines are uniquely determined by their input points, so the result is not implementation dependent.

The integrand of equation (3.17) thus becomes a piecewise 3^{rd} order exponential polynomial. The integral functions are 3^{rd} order exponential polynomials themselves, so the integral of each subinterval $[t_i, t_{i+1}]$ can be computed by an explicit formula. This outperforms numeric integration both in terms of speed and precision.

3.4.1.1 k_{3r}-Correction

For k_{3r}-correction, the right hand side of equation (3.9) must be computed. This is just the above problem with $k_{2r} + k_{3r}$ in the place of k. C_{rm} comes as a value table for $t_1, \ldots, t_N$. So C_r starts as a copy of the table. Then the integral is evaluated as above, giving another value table, which is added to C_{rm} to make the 2^{nd} approximation. So the whole iteration is in terms of N-vectors, and a new cubic spline is computed in every iteration. The cost is negligible since k_{3r}-correction is performed only once for every PET scan. It is not part of the voxel based procedure (COLOGNE method or NLS) that follows.

By equation (3.9), it is clear that the series of vectors should be growing in every iteration. The total increase (obtained by summation of the components) is monitored in every iteration, and break-off is triggered as soon as that increase fails to be positive. The maximum number of iterations is set to 100, but that limit was never reached. For numeric validation of the procedure, see section 6.4.

3.4.2 Finding the optimal k

As described in section 3.3, w_k is computed for different k's in order to find the k that maximizes correlation with the left hand side of (3.13). As it is clear from the previous section, computing w_k involves N evaluations of a 3^{rd} order exponential polynomial (the spline fitting need not be repeated for every k). That makes it fast enough to be called on-the-fly. So we implement a function computing the correlation coefficient from k. Any one-dimensional maximizer (such as the golden section method) can then be used to find the maximum of that function. Computation speeds of more than 1000 voxels per second could thus be reached, which is enough for the intended purpose.

If speed does become an issue (as it would be the case if numeric integration were used in 3.4.1), there is still the option of tabulating $w_k(t_i)$ for i=1,...,N and for a range of k's distributed over a suitable domain. Then for every voxel, correlation coefficients must be computed between $\frac{C_t}{C_r}$ and each tabulated w_k. Thus while saving the time for recomputing w_k, one might run into other expenses if the number of tabulated k's is large. Reducing their number leads to a tradeoff between speed and precision, since obviously the optimization result is one of the tabulated k's.

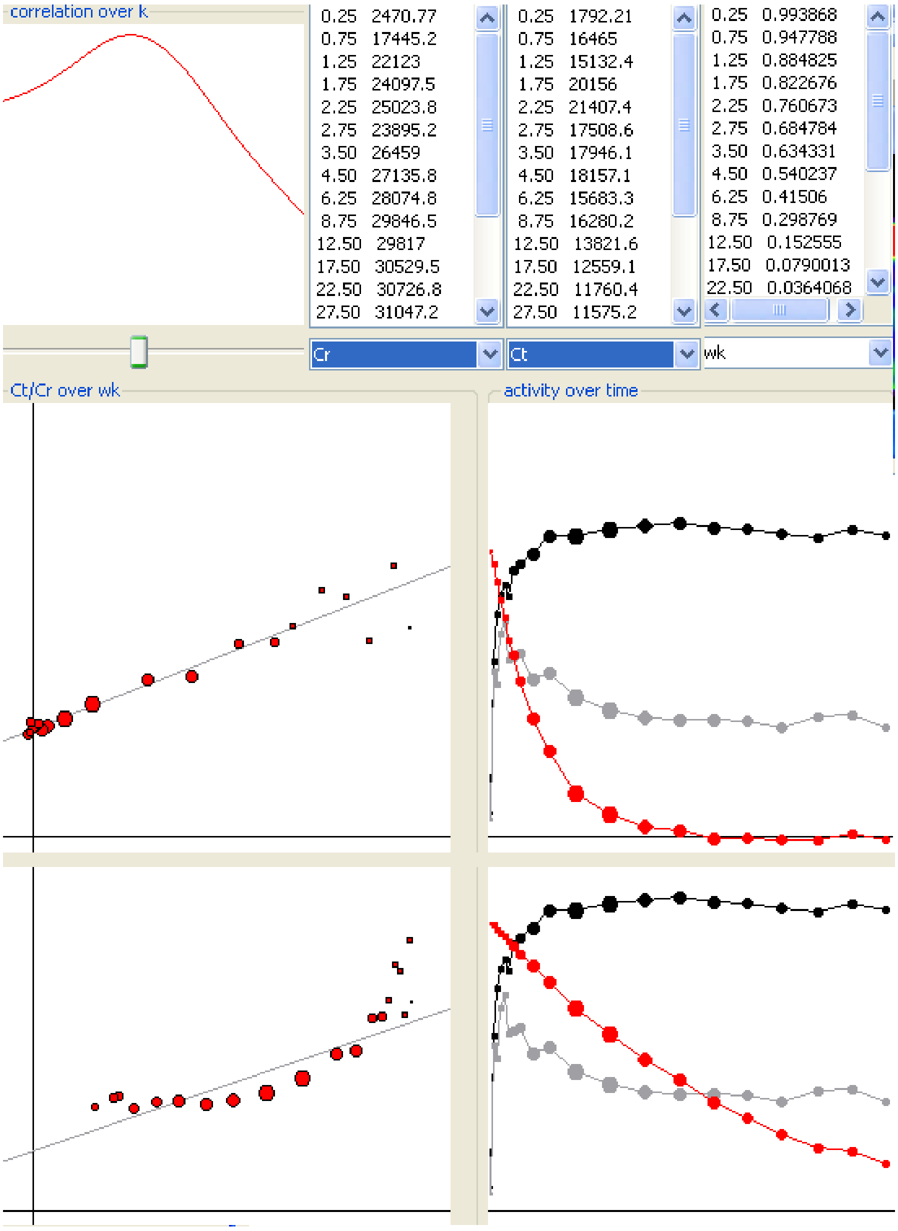
Before deciding to sacrifice precision for speed, there are three possible remedies to consider.

1. An equidistant distribution of the tabulated k's is *not* a good idea. In the end, we wish to discern $k_3 = 0.05$ clearly from $k_3 = 0.07$, but in high k_3 regions we care little whether it is 0.4 or 0.42. This calls for using smaller steps in lower areas. This can be achieved, for instance, by tabulating a geometric sequence of k's.

2. After finding the optimum tabulated k, an additional interpolation step can recover much of the sacrificed precision. However noisy the image data may have been to begin with, the function mapping the k's onto their correlations is smooth and analytic. Whereever its second derivative is not too close to zero, such a function looks — on a microscopic scale — like a parabola. This makes it possible to fit a parabola through the best tabulated k and its two neighbors, and determine the abscissa of its apex. This gives a compact formula interpolating the tabulated k's and approximating the optimum much closer.

3. The third idea is a bit risky since it relies on the target function having no more than one interior local maximum. In this case, we need not scan all tabulated k's in order to find the best. Instead, we can use a technique resembling binary search, bringing down the number of scans substantially. The details of the method can be worked out easily by anyone who has confidence in the above condition. On real images, we found it to hold true "almost everywhere".

In this work, an implementation based on 200 tabulated k's and all the tricks listed above was used. It reached a speed of more than 50,000 voxels per second while computing k_3 to a precision of 3 to 4 digits. Precision was checked by comparison with the golden section implementation.



```
0.25   2470.77
0.75   17445.2
1.25   22123
1.75   24097.5
2.25   25023.8
2.75   23895.2
3.50   26459
4.50   27135.8
6.25   28074.8
8.75   29846.5
12.50   29817
17.50   30529.5
22.50   30726.8
27.50   31047.2
```

Cr

```
0.25   1792.21
0.75   16465
1.25   15132.4
1.75   20156
2.25   21407.4
2.75   17508.6
3.50   17946.1
4.50   18157.1
6.25   15683.3
8.75   16280.2
12.50   13821.6
17.50   12559.1
22.50   11760.4
27.50   11575.2
```

Ct

```
0.25   0.993868
0.75   0.947788
1.25   0.884825
1.75   0.822676
2.25   0.760673
2.75   0.684784
3.50   0.634331
4.50   0.540237
6.25   0.41506
8.75   0.298769
12.50   0.152555
17.50   0.0790013
22.50   0.0364068
```

wk

Figure 3.1: *Visualizing the mechanism of COLOGNE. The right part shows the TACs C_t (gray) and C_r (black). The size of the dots corresponds to the weighting. The red curve w_k is computed from C_r and an estimate of k. On the left side, w_k is plotted on the abscissa, C_t/C_r on the ordinate, making a cloud of points. Changing k affects w_k and, hence, the cloud. The optimum k is 0.1749, it maximizes the correlation at 0.9760 (center). From the parameters of the regression line, k_2=0.1007 and k_3=0.0742 are computed. Underestimating k moves w_k up, shifting the dots to the right side, where they end up in a convex shape whose correlation is 0.8225 (bottom). k_2 and k_3 computed from the corresponding regression line would be 0.0213 and 0.0100.*

Chapter 4

Nonlinear Least Squares

In this chapter, reference based MP4A kinetic analysis is formulated as a weighted Nonlinear Least Squares (NLS)-problem. We then describe the implementation of a fast dedicated solver. As a spinoff, we obtain additional data which can be used for image analysis and error estimation, this will be addressed in section 4.7.

4.1 Specification

Recall that the kinetic model, given an input curve C_{pl} and kinetic constants k_1,k_2,k_3, defines C_t as spelled out in equation (3.4). In the reference based approach, C_{pl} is replaced by a reference curve C_r which can be sampled from the PET data. C_r is thought to be a scaled integral function of C_{pl}. Hence we need to reformulate equation (3.4) to have it depend on C_r instead of C_{pl}, this will be equation (4.11). As a result of the change, k_1 can no longer be determined in absolute terms, but as a fraction of k_{1r}, which is the k_1 of the reference region. So we set

$$q_1 := \frac{k_1}{k_{1r}} \tag{4.1}$$

and use q_1 consistently instead of k_1.

Now we have a function $\mathcal{F}$ defined by equation (4.11) mapping triples $(q_1 \mid k_2 \mid k_3)$ of kinetic constants to $\mathbb{R}^N$ where N is the number of frames. Given a measured TAC C_t, the task is to find a triple f' that minimizes

$$\sum_{i=1}^{N} w_i \cdot (C_t(t_i) - \mathcal{F}(f')(i))^2 \tag{4.2}$$

which is the sum of weighted squared differences between the measurement and the theoretic Time Activity Curve of every frame. For reasons given in section 2.7.3, we use the Decay Weights w_i defined in equation (2.7).

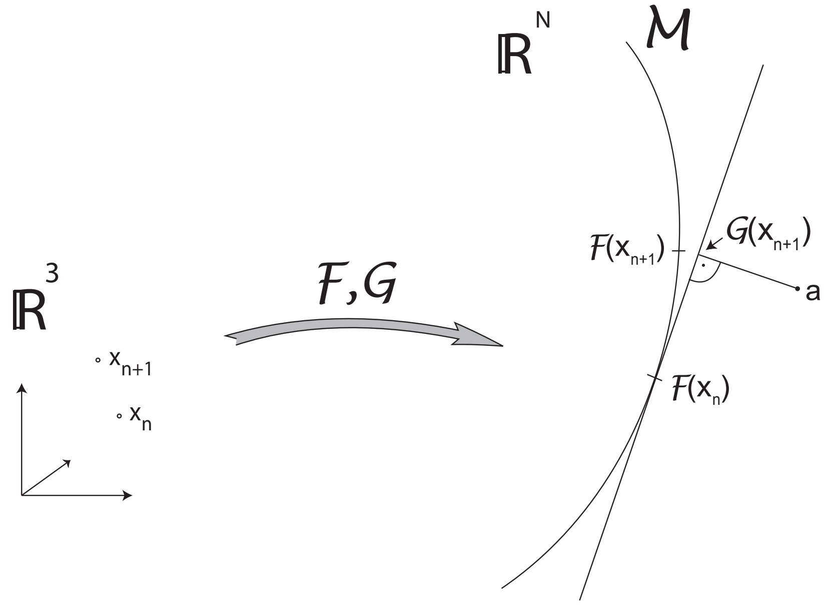


Figure 4.1: *An iteration of the Gauss Newton Method*

4.2 $\mathcal{F}$ and $\mathcal{M}$

Using $\mathcal{F}$ of equation (4.11), we define

$$\mathcal{M} := \mathcal{F}(\mathbb{R}^3) \tag{4.3}$$

It will be called the **Set of Model Compliant Curves**. A point p=$\mathcal{F}(p')$ of $\mathcal{M}$ then is the **Model Curve of p'**. It will be called **regular** if the partial derivatives of $\mathcal{F}$ in p' with respect to q_1, k_2, k_3 are linear independent. $\mathcal{M}$ resembles a 3-dimensional manifold embedded in $\mathbb{R}^N$, but as we shall see in section 4.6.2, not all of its points are regular.

Note that both the framing and C_r in its interpolated version are part of the definition of $\mathcal{F}$ and therefore $\mathcal{M}$. Hence we have introduced a notion of model compliance that depends on all these details.

4.3 Gauss Newton Method

This is a standard NLS solver for overdetermined nonlinear equation systems. From Newton solvers for general minimization problems, it is distinguished by not requiring second derivatives. We explain the method in the context of its intended application.

The situation is sketched in Figure 4.1. $\mathbf{a}$ is the measured TAC C_t that we seek to approximate. $\mathbf{x_n}$ is the n^{th} iteration point, a triple of kinetic constants. $\mathcal{F}(x_n)$ is its Model Curve, a point of $\mathcal{M}$. x_{n+1} is determined such that it solves the weighted least squares problem for the function $\mathcal{G}$ that parametrizes the tangential space of $\mathcal{M}$ about $\mathcal{F}(\mathbf{x_n})$:

$$\begin{aligned} \mathcal{G} : \mathbb{R}^3 &\to \mathbb{R}^N \\ \mathbf{x} &\mapsto \mathcal{F}(\mathbf{x_n}) + \mathbf{J_{x_n}}\mathcal{F} \cdot (\mathbf{x} - \mathbf{x_n}) \end{aligned} \tag{4.4}$$

where $J_{\mathbf{x_n}}\mathcal{F}$ is the Jacobi Matrix of $\mathcal{F}$ at $\mathbf{x_n}$. So we minimize the weighted squares of the residual vector $\mathcal{G}(\mathbf{x_{n+1}}) - \mathbf{a}$ where $\mathcal{G}$ is a linear mapping[1] whose definition depends of $\mathbf{x_n}$.

4.3.1 The linear Step

We seek $\mathbf{x}$ that yields the best approximation $\mathcal{G}(x) \approx \mathbf{a}$ in the weighted least squares sense. Let $\Delta\mathbf{x} := \mathbf{x} - \mathbf{x_n}$ denote the **shift vector**. The Jacobi matrix $J_{\mathbf{x_n}}\mathcal{F}$ has N rows and 3 columns:

$$J_{\mathbf{x_n}}\mathcal{F} = \begin{bmatrix} \frac{\partial f_1}{\partial q_1}(\mathbf{x_n}) & \frac{\partial f_1}{\partial k_2}(\mathbf{x_n}) & \frac{\partial f_1}{\partial k_3}(\mathbf{x_n}) \\ \frac{\partial f_2}{\partial q_1}(\mathbf{x_n}) & \frac{\partial f_2}{\partial k_2}(\mathbf{x_n}) & \frac{\partial f_2}{\partial k_3}(\mathbf{x_n}) \\ . & . & . \\ . & . & . \\ \frac{\partial f_N}{\partial q_1}(\mathbf{x_n}) & \frac{\partial f_N}{\partial k_2}(\mathbf{x_n}) & \frac{\partial f_N}{\partial k_3}(\mathbf{x_n}) \end{bmatrix} \tag{4.5}$$

Abbreviating $J_{\mathbf{x_n}}\mathcal{F}$ by $\mathbf{J}$, we have

$$\begin{aligned} \mathcal{F}(\mathbf{x_n}) + \mathbf{J}(\boldsymbol{\Delta}\mathbf{x}) &\approx \mathbf{a} \\ \mathbf{J}(\boldsymbol{\Delta}\mathbf{x}) &\approx \mathbf{a} - \mathcal{F}(\mathbf{x_n}) \end{aligned}$$

That makes $\Delta\mathbf{x}$ the least squares solution of an overdetermined linear equation system. Let $\mathbf{W}$ denote the diagonal matrix whose entries are the weights. Then, according to linear theory, the shift vector of the least squares solution is obtained by solving the system of the so called **normal equations**:

$$(\mathbf{J^TWJ})\boldsymbol{\Delta}\mathbf{x} = \mathbf{J^TW}(\mathbf{a} - \mathcal{F}(\mathbf{x_n})) \tag{4.6}$$

Its matrix $\mathbf{J^TWJ}$ is guaranteed to be regular as long as all weights are positive and the columns of $\mathbf{J}$ are linear independent, which is the case in every regular point of $\mathcal{M}$.

[1] more precisely: affine

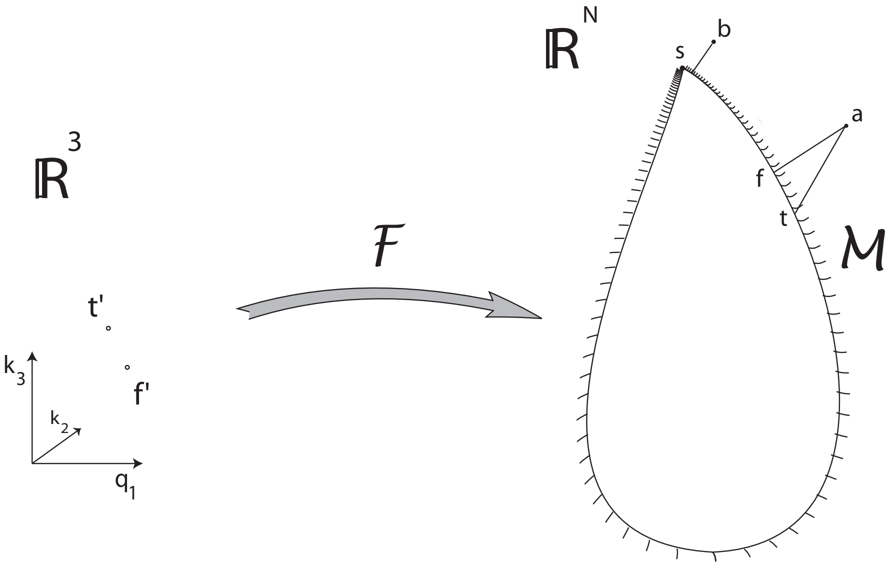


Figure 4.2: *Measured TAC, truth and NLS result*

4.3.2 Geometric Interpretation

$\mathbb{R}^N$ is the space of all TACs that might be measured. Based on the weighting $w_1, \ldots, w_N$, we define the inner product:

$$\langle \mathbf{u}, \mathbf{v} \rangle := \sum_{i=1}^{N} u_i w_i v_i \tag{4.7}$$

It induces the norm:

$$\|u\| = \sqrt{\sum_{i=1}^{N} u_i^2 w_i} \tag{4.8}$$

which in turn defines a metric of $\mathbb{R}^N$. Minimizing the weighted squares of $\mathcal{G}(\mathbf{x_{n+1}}) - \mathbf{a}$ comes down to finding the uniquely determined point of $\mathcal{G}(\mathbb{R}^3)$ which is closest to $\mathbf{a}$ in this metric. By construction (equation 4.4), $\mathcal{G}(\mathbb{R}^3)$ is the tangential space of $\mathcal{M}$ in $\mathcal{F}(\mathbf{x_n})$ and, in the vicinity of $\mathcal{F}(\mathbf{x_n})$, its parametrization matches the parametrization of $\mathcal{M}$ by $\mathcal{F}$. Therefore, $\mathcal{G}(\mathbf{x_{n+1}})$, which is the closest point of the tangential space, will not be far away from $\mathcal{F}(\mathbf{x_{n+1}})$, which is therefore expected to improve on the previous iteration point, although this cannot be guaranteed in the nonlinear situation.

In **Figure 4.2** the Gauss Newton method has finally converged and found **f** as the closest point of $\mathcal{M}$ to **a**. The reason why **a** itself is not a point of $\mathcal{M}$ is that it is disfigured by noise. Assuming correctness of the model, it may originally have been located at **t**, from where the noise carried it to **a**, from where NLS obtained **f** as the foot of a perpendicular dropped onto $\mathcal{M}$. **t** and **f** have preimages **t'** and **f'** under $\mathcal{F}$, where **t'** are the "true" kinetic constants, and **f'** is the estimate found by NLS.

One may look upon $\mathcal{M}$ as carrying a 3-dimensional coordinate system, hinted by the (one-dimensional) tickmarks in Figure 4.2. Then **f'** is found by reading the coordinates at **f** where the perpendicular hits $\mathcal{M}$. It is obvious that precision will suffer if **f** is at a location where the tickmarks are close. This is the case for TACs like **b**, which are close to a singular point **s** of $\mathcal{M}$. Such a situation will be identified in section 4.6.2. Quantitative treatment, leading to an error estimate for k_3, is the subject of chapter 8.

4.4 Model Function and Derivatives

Now for the definition of $\mathcal{F}$. According to (3.13), we have

$$C_t(t) = C_r(t)\,[Aw_k(t) + B]$$

...using formula (3.16) for w_k ...

$$= C_r(t)\left[A - Ak \cdot \frac{\int\limits_0^t C_r(s)\cdot e^{k(s-t)}ds}{C_r(t)} + B\right]$$

$$= C_r(t)(A+B) - Ak\int\limits_0^t C_r(s)\cdot e^{k(s-t)}ds$$

where, by (3.15) and (3.10),(3.11)

$$A + B = \frac{k_1}{k_{1r}}$$

$$Ak = \frac{k_1}{k_{1r}}(1-q)k = \frac{k_1 k_2}{k_{1r}}$$

So we have

$$C_t(t) = \frac{k_1}{k_{1r}}C_r(t) - \frac{k_1 k_2}{k_{1r}}\int\limits_0^t C_r(s)\cdot e^{k(s-t)}ds \tag{4.9}$$

It is now expressed in terms of C_r instead of C_{pl}. As a consequence, we have k_{1r} in the formula and can now replace $\frac{k_1}{k_{1r}}$ with q_1 as advertised in equation

(4.1). The i'th component function f_i of $\mathcal{F}$ takes three kinetic constants as arguments and returns C_t at time t=t_i:

$$f_i\left(q_1 \middle| k_2 \middle| k_3\right) := q_1 \left[C_r(t_i) - k_2 \int\limits_0^{t_i} C_r(s) \cdot e^{(k_2+k_3)(s-t_i)} ds \right] \tag{4.10}$$

and $\mathcal{F}$ is composed of the f_i:

$$\mathcal{F}\left(q_1 | k_2 | k_3\right) := \left(f_1\left(q_1 | k_2 | k_3\right), \ \dots \ , f_N\left(q_1 | k_2 | k_3\right)\right) \tag{4.11}$$

Next, we need the partial derivatives of the f_i with respect to the kinetic constants, for they make up the Jacobi matrix of $\mathcal{F}$. Looking at equation (4.10), it is obvious that the derivative with respect to the first argument is the contents of the square brackets:

$$\frac{\partial f_i}{\partial q_1} = C_r(t_i) - k_2 \int\limits_0^{t_i} C_r(s) \cdot e^{(k_2+k_3)(s-t_i)} ds \tag{4.12}$$

k_3 occurs only in the exponent of (4.10), so by partial differentiation under the integral sign we get a factor $(s - t_i)$ inside the integral:

$$\frac{\partial f_i}{\partial k_3} = -q_1 k_2 \int\limits_0^{t_i} C_r(s) \cdot (s - t_i) \cdot e^{(k_2+k_3)(s-t_i)} ds \tag{4.13}$$

k_2 is found in the same position as k_3 and also in front of the integral, so by the product rule we obtain the above plus another copy of the integral:

$$\frac{\partial f_i}{\partial k_2} = -q_1 k_2 \int\limits_0^{t_i} C_r(s) \cdot (s - t_i) \cdot e^{(k_2+k_3)(s-t_i)} ds - q_1 \int\limits_0^{t_i} C_r(s) \cdot e^{(k_2+k_3)(s-t_i)} ds \tag{4.14}$$

4.5 Implementation and Complexity

4.5.1 $\mathcal{F}$ and Derivatives

For abbreviation, let $k = k_2 + k_3$. Equations (4.10) to (4.14) contain just two integral types, namely

$$\text{comp}_1(t) := \int\limits_0^{t} C_r(s) \cdot e^{k(s-t)} ds \tag{4.15}$$

and

$$\mathrm{comp}_2(t) := \int\limits_0^t (s-t) \cdot C_r(s) \cdot e^{k(s-t)} ds \tag{4.16}$$

where t occurs both as an integration limit and as part of the integrand. But this is easily disentangled. Letting

$$\mathrm{int}_1(t) := \int\limits_0^t C_r(s) \cdot e^{ks} ds \tag{4.17}$$

and

$$\mathrm{int}_2(t) := \int\limits_0^t s \cdot C_r(s) \cdot e^{ks} ds \tag{4.18}$$

we have

$$\mathrm{comp}_1(t) = e^{-kt} \cdot \mathrm{int}_1(t) \tag{4.19}$$

and

$$\begin{aligned} \mathrm{comp}_2(t) &= -t \cdot \mathrm{comp}_1(t) + e^{-kt} \cdot \mathrm{int}_2(t) \\ &= e^{-kt} \left(-t \cdot \mathrm{int}_1(t) + \mathrm{int}_2(t)\right) \end{aligned}$$

C_r is a natural cubic spline changing formula in $t_1,\ldots,t_N$. Computation of $\mathrm{int}_1(t_i)$ and $\mathrm{int}_2(t_i)$ therefore comes down to adding up i integrals of third order and fourth order exponential polynomials, respectively. These are exponential polynomials themselves: one easily verifies that

$$\begin{aligned} \int e^{ks} ds &= \frac{1}{k} \cdot e^{ks} \qquad (4.20) \\ \int s \cdot e^{ks} ds &= \frac{ks-1}{k^2} \cdot e^{ks} \\ \int s^2 \cdot e^{ks} ds &= \frac{(ks)^2 - 2ks + 2}{k^3} \cdot e^{ks} \\ \int s^3 \cdot e^{ks} ds &= \frac{(ks)^3 - 3(ks)^2 + 6ks - 6}{k^4} \cdot e^{ks} \\ \int s^4 \cdot e^{ks} ds &= \frac{(ks)^4 - 4(ks)^3 + 12(ks)^2 - 24ks + 24}{k^5} \cdot e^{ks} \end{aligned}$$

Computation of the enumerator polynomials by Horner's scheme takes r additions and multiplications, where r is the polynomial order. The integral function

$$\int (a_4 s^4 + a_3 s^3 + a_2 s^2 + a_1 s^1 + a_0) \cdot e^{ks} ds \tag{4.21}$$

of a fourth order exponential polynomial will therefore take 4+3+2+1+4=14 additions and multiplications, where one of the enumerators in (4.20) is evaluated for each summand of (4.21) and 4 more additions and multiplications are used to involve the outer coefficients: $\frac{a_4}{k^5}, \frac{a_3}{k^4}$ etc. and add up the terms. For a third order exponential polynomial the cost is 3+2+1+3=9 additions and multiplications. Joining the e^{ks} factors requires two more multiplications and one exponentiation. The integral function of a fourth order exponential polynomial thus comes to (14 | 16 | 1) additions / multiplications / exponentiations, for third order we have (9 | 11 | 1). A definite integral requires 2 evaluations of the integral function and one subtraction, making a total cost of (29 | 32 | 2) additions / multiplications / exponentiations for fourth and (19 | 22 | 2) for third order.

$\text{int}_1(t_i)$ and $\text{int}_2(t_i)$ are as many sums of definite integrals as C_r is composed of different formulae: for t=t_i there are i formulae involved. $\text{int}_2(t_i)$ therefore costs $(29i\,|\,32i\,|\,2i)$ and $\text{int}_1(t_i)$ $(19i\,|\,22i\,|\,2i)$ additions / multiplications / exponentiations. We need value tables of int_1 and int_2 for $t_1, \ldots, t_N$. Since the entries differ only by their upper integration limit, they can be computed incrementally. The added cost of both integral tables is then the same as for only $\text{int}_1(t_N)$ and $\text{int}_2(t_N)$, minus the two exponential terms that are identical in both tables: $(48N \mid 54N \mid 2N)$ additions / multiplications / exponentiations.

We then compute tables for comp_1 and comp_2 using formulae (4.19), and, based on these, the values of $\mathcal{F}$ and its partial derivatives by formulae (4.10), (4.12),(4.13),(4.14), in $\mathcal{O}(N)$ time, filling the Jacobi matrix. From it, we compute the 3x3 matrix $\mathbf{J^T W J}$ and the right hand side of equation (4.6). Each entry of $\mathbf{J^T W J}$ is an inner product of type (4.7) of two columns of J, so $\mathbf{J^T W J}$ it is computed in linear time as well. The same applies to the right hand side of the equation system. Solving it is possible in constant time. So the cost for an iteration of the Gauss Newton Method is $\mathcal{O}(N)$.

In Chapter 5, we shall restrict the number of iterations to a maximum, giving linear complexity to a single k_3 evaluation. To compute a parametric image, it must be taken times the number of voxels.

4.5.2 Spline Interpolation of C_r

A natural cubic spline interpolating y_1, y_2,...,y_N at abscissae x_1, x_2,...,x_N can be written in the form

$$Spline(x) = a_0 x + b + \sum_{i=1}^{N} a_i \underline{(x - x_i)^3} \tag{4.22}$$

where the underline means that all negative values are replaced with 0. Then there are N equations to meet the interpolation conditions and 2 additional degrees of freedom originating from a_0 and b. These are used to ensure that the polynomial to the right of x_N is of 1^{st} order. The resulting (N+2)x(N+2) linear equation system was solved by Gaussian elimination. This was considered to be sufficiently precise since N, the number of frames, never exceeded 32 on real PET data, and the method was tested on 100 abscissae to ensure the spline satisfied the specification. Speed is not an issue since we need one cubic spline per image.

4.6 More on $\mathcal{F}$ and $\mathcal{M}$

4.6.1 Mathematical Footwork

$\mathcal{F} : \mathbb{R}^3 \rightarrow \mathbb{R}^N$ is defined by equation (4.11). Note that the framing times t_1,. . . ,t_N and the reference function C_r in its interpolated version, are part of its definition, and therefore, part of the definition of $\mathcal{M} = \mathcal{F}(\mathbb{R}^3)$. Some trivial observations on its structure are now provided, for reference in later sections.

Theorem 4.1 *$\mathcal{F}$ is linear in its first argument q_1, and also with respect to its parameter C_r.*

It means that $\mathcal{F}$ scales with both and is additive with respect to both. For proof, consider the component f_i (equation 4.10). It certainly is linear with respect to q_1. Now consider t_i as fixed, and investigate how f_i changes in response to C_r. The first summand is a linear function mapping C_r on the real number $C_r(t_i)$. For fixed k_2, k_3, the second summand is another real valued function of C_r. It is linear because integration is linear. So f_i is linear as it is the sum of two linear functions, and $\mathcal{F}$ is linear because all its components are.

Corollary 1 *If p is a point of $\mathcal{M}$, then so are its scalar multiples.*

Proof: Since p is on $\mathcal{M}$, it has a preimage p'=(q_1,k_2,k_3) under $\mathcal{F}$. Then because $\mathcal{F}$ is linear in its first argument, for every $\alpha \in \mathbb{R}$, ($\alpha \cdot q_1$,k_2,k_3) is a preimage to $\alpha \cdot p$. Hence $\alpha \cdot p$ is on $\mathcal{M}$.

Corollary 2 *If C_r is scaled by some factor $\alpha > 0$, its $\mathcal{M}$ will not change.*

Proof: By Theorem 1, $\mathcal{F}$ will also be scaled with α. Let $\mathcal{F}'$ be that scaled mapping and $\mathcal{M}'$ its manifold, both arising from αC_r. If p is on $\mathcal{M}$, $\frac{1}{\alpha} \cdot p$ is also on $\mathcal{M}$ by Corollary 1. Its preimage under $\mathcal{F}$ is mapped to p by $\mathcal{F}'$, hence p is on $\mathcal{M}'$, hence $\mathcal{M} \subset \mathcal{M}'$. The same argument applied to $1/\alpha$ shows that $\mathcal{M}' \subset \mathcal{M}$.

Theorem 4.2 *C_r is a linear function of C_{pl}*

This is evident by looking at its definition (3.5) and (3.6).

Theorem 4.3 *Consider a fixed reference curve C_r and framing schedule, with $\mathcal{F}$ and $\mathcal{M}$ defined by these. Consider a Time Activity Curve **a** and a real positive scalar c. Then if (q_1, k_2, k_3) is a weighted least squares solution for a, $(c \cdot q_1, k_2, k_3)$ is a weighted least squares solution for the scaled TAC ca.*

Proof: $\mathcal{F}(q_1, k_2, k_3)$ is a closest point of $\mathcal{M}$ to **a**, as measured with $\|.\|$ defined by equation (4.8). Observe that the distance

$$\|\mathcal{F}(\alpha \cdot q_1 \ , \ k_2 \ , \ k_3) - \alpha \cdot a\|$$

scales with α, as a consequence of Theorem 4.1. Now suppose there was a better solution $(j_1 \ , \ j_2 \ , \ j_3)$ to the approximation problem for c**a**. So there is

$$\|\mathcal{F}(j_1 \ , \ j_2 \ , \ j_3 \ ) - c \cdot a\| < \|\mathcal{F}(c \cdot q_1 \ , \ k_2 \ , \ k_3 \ ) - c \cdot a\|$$

Then by scaling both distances with $\alpha := \frac{1}{c}$ we obtain

$$\left\|\mathcal{F}\left(\frac{j_1}{c} \ , \ j_2 \ , \ j_3\right) - a\right\| < \|\mathcal{F}(q_1 \ , \ k_2 \ , \ k_3) - a\|$$

which is a contradiction to the optimality of $(q_1 \ , \ k_2 \ , \ k_3)$.

Theorem 4.4 *Consider a reference curve C_r, a TAC **a**, and a weighted least squares solution (q_1, k_2, k_3) of **a** against C_r. Then for every constant $\alpha > 0$, $\left(\frac{1}{\alpha} q_1, k_2, k_3\right)$ is a weighted least squares solution of **a** against αC_r.*

Proof: Consider $\mathcal{F}$ as arising from C_r by equation (4.11) and $\mathcal{F}'$ as arising from αC_r. Let (q_1, k_2, k_3) be a least squares solution for C_r. Then $p := \mathcal{F}(q_1, k_2, k_3)$ is a closest point to **a** on $\mathcal{M}$. By Corollary 2, $\mathcal{M}$ as arising from αC_r is the same. So p is still a closest point. Since by Theorem 4.1 $\mathcal{F}' = \alpha \mathcal{F}$, it maps $\left(\frac{1}{\alpha} \cdot q_1, k_2, k_3\right)$ onto p, hence $\left(\frac{1}{\alpha} \cdot q_1, k_2, k_3\right)$ is a least squares solution.

Theorem 4.5 *Scaling of C_{pl}, C_r or C_t with a factor $\alpha > 0$ will not change k_2 and k_3 of a weighted least squares solution.*

Proof: For C_t, this follows from Theorem 4.3. For C_r, it follows from Theorem 4.4. Scaling of C_{pl} results in scaling of C_r by Theorem 4.2.

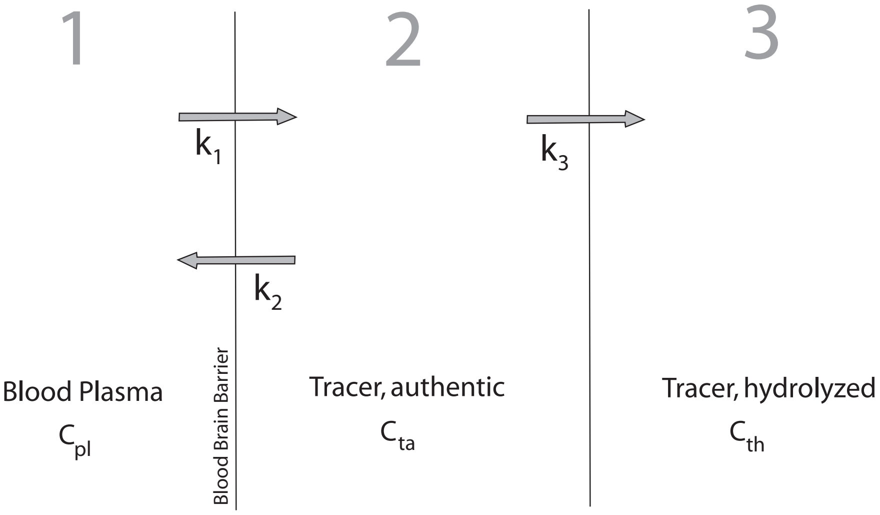


Figure 4.3: *The 3-Compartment Model*

4.6.2 Singular Points of $\mathcal{M}$

Consider again the compartment model (Figure 4.3) that led to equation (3.4). By Assumption 1, C_t is the signal coming from compartments 2 and 3 (authentic and hydrolyzed tracer in tissue). The reference curve arises from this model by equating k_3 with infinity. This implies that every molecule, once having reached compartment 2, is immediately transferred to compartment 3. As there is no way back, its radioactivity remains trapped in the tissue.

Now, consider what happens if k_2 is zero. Then there is no backflow from the tissue to the blood. Once having reached compartment 2, radioactivity is also trapped, although for a different reason. To the PET signal, it makes no difference, since it doesn't play a role if an ^{11}C atom decays as part of an authentic tracer molecule or as part of a metabolite.

Hence the model maps k_2=0 to the same curve as $k_3 = \infty$. Equation (4.10) reveals that this curve is a multiple of C_r, with q_1 as the scaling factor. Moreover, if k_2=0, k_3 has no bearing on the result. Thus, $\mathcal{F}$ maps all the points $(q_1 \mid 0 \mid k_3)$ onto the same point q_1C_r, which is a scaled version of the reference curve C_r. So the reference curve and all its scaled versions are singular points of $\mathcal{M}$.

Conversely, if the input curve C_t is almost a scaled version of C_r, there will be no way of knowing if the similarity is caused by large k_3, or by low k_2 (with arbitrary k_3). As triples with different k_3 are mapping to almost the same curve, reconstruction of the original k_3 from that curve gets impossible.

4.6.3 *simcr*

So we run into trouble when a TAC **a** approaches a scalar multiple of the reference curve C_r. We introduce a measure of similarity between the two curves: by a well-known formula, the cosine of the angle between them is

$$simcr(a) := \frac{\langle a, C_r \rangle}{\|a\| \cdot \|C_r\|} \tag{4.23}$$

It is also the projection of **a**, normalized to length 1, onto C_r. If that measure is 1, then **a** is a scalar multiple of C_r. Note that this definition depends on the framing (which comes in via the Decay Weights as they define the inner poduct (4.7)) and on C_r which must be sampled using a reference mask before *simcr* can be computed. Its advantage is that it does not require computing the kinetic constants first. Hence it is inexpensive, and, unlike with k_3, its computation never fails.
Having *simcr* close to 1 is an indication that k_3 will be hard to compute, so it increases the likelihood that NLS will fail.

simcr is indifferent to scaling of its argument, a property it shares with k_2 and k_3. Its theoretic codomain is the interval [-1,1], but negative values were never seen in practice. Typical values for TACs C_t of “nice” NLS performance range between 0.95 and 0.97. A parametric image of *simcr* is shown in Figure 13.2.

4.7 Image Modalities

are quantities that can be computed for every voxel to make parametric images. Examples are the kinetic constants themselves, and *simcr* as of section 4.6.3. The modalities to be defined in this section depend on the Model Curve **f** of the NLS result, so they require NLS to run first.

4.7.1 $noise_{abs}$

is the distance of the measured curve **a** from $\mathcal{M}$, as represented by its closest point **f** (see Figure 4.2):

$$noise_{abs} := \|\mathbf{a} - \mathbf{f}\| \tag{4.24}$$

The name slightly overstates its significance. On real image data, whatever offsets **a** from $\mathcal{M}$, should mostly by attributable to noise. But it can also be caused by ***bias***, a collective term for all systematic error allowed to creep in at any of the stages of Figure 1.3. An example is presented in section 12.1, where the kinetic model itself is to blame for the bias.

Even if all was noise, $noise_{abs}$ would just represent the larger part of it: the part orthogonal to $\mathcal{M}$. The "true" noise corresponds to the distance between **a** and **t**, rather than between **a** and **f**, and the part that escapes direct measurement is parallel to $\mathcal{M}$. But this is the more interesting part, as it accounts for the misassessment of k_3. It will be the subject of chapter 8 to build a bridge between the two.

$noise_{abs}$ is the quantity (or rather, its square root) that NLS minimizes. Hence it is available from an NLS run at almost zero cost. All there needs to be computed is a single extra evaluation of $\mathcal{F}$, since $\mathbf{f} = \mathcal{F}(f')$ where **f'** is the NLS result.

4.7.2 $noise_{rel}$

is obtained by dividing $noise_{abs}$ by the length of the vector **f** (Figure 4.2). If $noise_{abs}$ is the "noise", then the length of **f** is the "signal", the part that is model-compliant and carries the information about k_3. $noise_{rel}$ is the inverse **Signal-to-Noise Ratio** (**SNR**). On parametric images of $noise_{rel}$, the noisy spots of an image light up (Figure 12.1). Of course there always remains the question if it isn't bias rather than noise.

4.7.3 *framedev*

Like the previous, this modality also compares the measured TAC **a** with its Model Compliant counterpart **f** (Figure 4.2). But instead of focusing on distances, it contemplates the i^{th} frame in isolation.

$framedev_i$ is defined as the measured intensity in frame i, minus the intensity in frame i of the Model Curve **f** of the NLS result. Again **f** replaces **t**, which is unavailable. Recall that **t** is the original Model Curve and **a** its measured, noisy version, so $\vec{\mathbf{ta}}$ is the noise vector. If it really was pure noise, its i^{th} component should average out to zero when compared for many voxels. If it turns out otherwise, it is a signal for bias. Hence, parametric images of the *framedev* modalities are the next thing to check when a region lights up in the $noise_{rel}$ modality.

In a viewer calibrated to display positive values in red and negative values in blue, one expects to see a mix of red and blue voxels. On Gauss-filtered images, it becomes a patchwork of small red and blue areas (Figure 12.1). As long as there is a balance of red and blue and the areas are not too large and intense, they are not necessarily indicative of bias.

In section 12.1, the $framedev_2$ modality is used to identify a source of bias, and provide a mask for the region where that bias occurs.

4.7.4 *uff*

is a shorthand for the **"unfiltered feedback"** modality. It is a quality marker designed to validate image preprocessing (i.e., masking and filtering) that precedes k_3 computation, and can only be understood in the context of chapters 9 or 10. Filtering invites Partial Volume Effects, thus inducing bias to the final k_3 results. To assess this bias, we need something that can peek through the filtering step.

uff is computed from on an **unfiltered TAC u** that is sampled from a voxel of the normalized frames (section 9.2.2) *before* Gauss filtering, and the Model Curve **f** of the NLS result. Note that NLS cannot be invoked for **u** itself which is much too noisy. Instead, its input is **a**, the TAC sampled after preprocessing. **f** is compared with **u** to obtain "feedback". *uff* is a measure of similarity between the two, defined as follows:

$$uff := 1 - \frac{\|u - f\|}{\|u\|} \tag{4.25}$$

If **u** was noise free and fully explained by the NLS result (q_1, k_2, k_3), $\|u - f\|$ would be zero. The enumerator is the length of the "unexplained" part of **u**. If, say, 30% are unexplained, the fraction is 0.3 and *uff* is 0.7. So we can say "the computed triple explains 70% of **u**". For a "nice" voxel of a Sample 1 image, *uff*=0.7 is a typical value. In Sample 3 and 4 images that were generated with more smoothing during reconstruction, *uff* was between 0.8 and 0.9.

Interpretation of the marker is problematic since there are two complicating factors:

1. The marker measures similarity of TACs. They can be similar to each other even if the kinetic constants aren't, in particular this happens near the singular points of $\mathcal{M}$ (see section 4.6.2). This provides for high values of *uff* whenever *simcr* is large and NLS finds *any* plausible explanation, be it by low k_2, high k_3 or both.

2. Another issue is its sensitivity to noise. Noisy curves are not matched by *any* Model Compliant curve, so they make *uff* drop. High $noise_{rel}$ imply low readings of *uff*. In Table 12.2 *uff* has been listed alongside $noise_{rel}$ in order to document this effect. It explains the increase of *uff* with the zonal index in Tables 10.7 and 10.5.

Interference patterns were seen on *uff* images of Sample 1. They are caused by resampling of images during coregistration/normalization. Resampling by trilinear voxel interpolation leads to noise reduction on certain voxels but not on others, forming an interference pattern that is made visible by the noise sensitivity of *uff*.

4.7.5 *iter*

is the number of iterations of an NLS run. It appears in some tables of chapters 12 and 13, and gives nice parametric images (Figure 13.2).

4.7.6 Other Modalities

k3diff is k_3 computed by COLOGNE minus k_3 computed by NLS. ***k3var*** and ***k3dev*** are error estimates for k_3, they are introduced in chapter 8.

Chapter 5

Calibrating the Solver

While the focus of the previous chapter was on efficiency of the iteration step, we are now turning toward convergence properties, adressing such issues as the choice of starting point and break-off rule, and possible modifications.

5.1 Properties of the Gauss Newton Method

Gauss Newton methods are discussed at length in [9, 10]. They belong to the larger family of Newton-like methods. The relation to the Newton method proper is most obvious in the special case where the number of unknowns matches the dimensionality N of the measured data vector **a** and hence, the Jacobi matrix J of equation (4.5) is quadratic. It is then reasonable to expect an exact, rather than approximative, solution x to the problem

$$\mathcal{F}(x) = a \tag{5.1}$$

If J is regular and all weights are positive, the Gauss Newton step (4.6) then simplifies to

$$J \cdot \Delta\mathbf{x} = \mathbf{a} - \mathcal{F}(\mathbf{x_n}) \tag{5.2}$$

which happens to be a step of the standard Newton method for solving the nonlinear equation system (5.1).

In the given situation where N exceeds the number of unknowns, Gauss Newton can be compared to the standard Newton method of minimizing the residual squares

$$R(x) := \sum_{i=1}^{N} w_i \cdot (C_t(t_i) - \mathcal{F}(x)(i))^2 \tag{5.3}$$

It searches for a point that has all first order derivatives of R disappear, and in doing so, makes use of the second order derivatives of R, requiring computation of its Hesse Matrix. As with Newton methods in general, we can expect local quadratic convergence.

The advantage of Gauss Newton is that it does not require second derivatives. Its relation to the standard Newton method has been explored extensively in [9]. The Gauss Newton iteration matrix $J^T W J$ of equation (4.6) is related to the Newton iteration matrix G via

$$G = J^T W J + S \tag{5.4}$$

where

$$S = \sum_{i=1}^{n} r_i \cdot H_i \tag{5.5}$$

where r_i are the weighted residuals $\sqrt{w_i} \cdot (C_t(t_i) - \mathcal{F}(x)(i))$ and H_i their respective Hesse matrices. The simplification made by Gauss Newton thus consists in omitting S from (5.4). Consider, in equation (5.5), S at a minimizer x^* of R. The better the approximation, the smaller are the r_i and hence the error allowed by omitting S. The same can be said if the r_i (or equivalently, $\mathcal{F}$) are almost linear in the vicinity of the minimizer, and hence the entries of the H_i are small. If R is zero at the minimizer, [9] shows that Gauss Newton will be of local quadratic convergence. Otherwise, convergence is only linear, and its speed decreases as the residual size or the nonlinearity of the problem increases, making S larger compared to $J^T W J$. Eventually, there will be no convergence at all, not even locally.

5.1.1 Modifications proposed in Literature

[9] goes on to investigate these cases and finds that the Gauss Newton steps are always in a direction of descent as long as J has full column rank (which is the case in every regular point of $\mathcal{M}$). The reason why they lead to divergence is that they are too long. This suggests that the problem of divergence can be fixed by truncating Gauss Newton steps where necessary in order to ensure descent. Such modifications are referred to as "damped Gauss Newton" in [9].

Another idea has been introduced by Marquardt and Levenberg [27]. It modifies the iteration matrix, where necessary, by adding a multiple of the unit matrix. This causes it to be positive definite and regular even if it wasn't so to begin with, and rotates the direction of search toward the steepest descent while also cutting down on step length. Marquardt-Levenberg methods are particularly well suited in general Newton minimization procedures that could otherwise be attracted by saddle points, where the Hesse matrix has eigenvalues of either sign. In Gauss Newton, the iteration matrix $J^T W J$ is positive semidefinite by construction, and positive definite in every regular point of $\mathcal{M}$.

5.2 Alternatives regarding the Starting Point

The following options were considered at the outset:

1. Start NLS from the result of a neighboring voxel
2. Start NLS from a COLOGNE result computed for the same voxel
3. Use a fixed starting point for all voxels

(1) seems to carry the promise both of speed and safety: as the kinetic constants of neighboring voxels ought to be similar, we can expect to start in the attractor range of the global optimum and reach the target in a few iterations. Its drawback is that it creates inter-voxel dependencies, so we need to decide on an order in which to process the voxels. Second, we might just as well get "attracted" by a suboptimal trail and follow that over a distance of many voxels. Of course, there are endless variations of how the policies might be refined.

(2) seemed to be the obvious choice since the COLOGNE method is fast and had already been implemented, so one can only gain by allowing NLS to improve on its results. However,

(3) was adopted after it had turned out to be feasible, using S_1:=(1;0.1;0.1) as the universal starting point. It benefits from simplicity and providing a standalone algorithm, which can be validated in isolation and later compared with COLOGNE.

5.3 Convergence Properties, Break-off Rule, Acceptance and Failure

In experiments with real PET data (see appendix, A.2) our implementation appeared to converge at linear speed, reducing the distance to the optimum by $\frac{1}{7.28}$ per iteration on average in cortex, and by $\frac{1}{4.84}$ in Hippocampus. Some other voxels converged more slowly. We therefore devised the following break-off policy:

1. Break-off is based on the length $\|x_{n+1} - x_n\|$ of every iteration step
2. We break off if it falls below a threshold l_1.
3. The maximum number of iterations allowed is n_1.
4. If (2) has not been triggered after n_1 iterations, we apply a larger threshold l_2 to the last step, and, based on it, accept the result or report failure.

n_1, l_1 and l_2 were set to 20, 10^{-7} and 10^{-4}, based on our specific needs and experiments explained in appendix A.2 and A.2.1. The number of voxels that fail under this regime ranges between 5 and 15 percent of the brain, depending on the individual. The loss appears necessary: at least in the reference regions we cannot expect results for principal reasons, and they are accompanied by other regions of prohibitively high k_3. It therefore seems realistic not to "force" convergence by any means, rather allow for failure, which eliminates unreasonable results.

5.4 Step Size Control

has been mentioned in section 5.1.1 as a remedy in situations where unmodified Gauss Newton is divergent. At first, we saw no reason to employ it since convergence properties appeared satisfactory on real PET data. Later we discovered cases of divergence with simulated data (appendix A.3). We then restricted the method to the first octand, in the following way:

A parameter λ is introduced, it was usually set to 0.9. An iteration is then allowed to run 90% of the way, but no further, towards any of the 3 border planes defined by q_1=0, k_2=0, k_3=0. Any approach beyond this point must wait for the following iteration, and crossing the border is entirely impossible. Setting λ to 0 turns the feature off.

We thus picked up the idea of "damped Gauss Newton" (section 5.1.1), however replacing line-search as proposed in [9] by a much cheaper way to determine the truncated step size, which is motivated by assuming that results must be positive[1]. As aforementioned problematic cases responded well and adverse effects were not observed, we adopted λ=0.9 as a general policy.

5.5 Influence of the Starting Point

While S_1=(1;0.1;0.1) remains our universal starting point, it needs to be demonstrated - as a consistency check - that results are not biased by this choice. In experiments with the data of subject M01014, we also ran the algorithm from S_2:=(0.5;0.1;0.1) and S_3=(1;0.05;0.05) and compared k_3 results with those obtained from S_1. Comparing S_1 and S_2, the largest k_3 difference seen in all voxels was 0.00047. Voxels requiring less than 20 iterations were much more closely matched owing to the break-off policy, having mostly differences below 10^{-6}. Runs from S_2 took, on average, 0.7 iterations more to converge than runs from S_1.

Comparison between S_1 and S_3 runs showed that the latter required, on average, 0.8 iterations more. Differences in k_3 results fell below 0.0007 for

[1] negative q_1, k_2 or k_3 make no sense in the context of the biological model

all but 291 voxels, where they ranged between 0.0614 and 1.528. In terms of residual weighted squares ($noise_{abs}$ modality of section 4.7.1), the S_1 result was better in 237 of these cases and the S_3 result in the 54 other cases. Of the latter, the maximum improvement of weighted squares was 4.6% with a mean of 1.3%. Thus, S_1 led to a suboptimal solution in about 0.025 percent of all non-failing voxels. However, the S_3 result averaged $(k_2 | k_3) = (0.0066 | 0.0140)$, with k_2 being lower than anything expected in brain, so these 54 voxels are a population of extreme outliers.

5.6 Suboptimal Solutions

Apart from the above example, we also hunted for suboptimal NLS solutions by comparing them with COLOGNE results, using the PET data of subject M01002. Of 221000 voxels evaluating successfully with both methods, 239 NLS results were found to be suboptimal. They averaged $(k_2 | k_3) = (0.0643 | 0.153)$ with NLS and $(0.547 | 0.600)$ with COLOGNE. This time, $k_2 = 0.547$ is higher than anything expected in brain. NLS could be made to find the COLOGNE result in 174 of the 239 cases by starting it from S_4=(1;0.8;0.3), but this point is even further away from k_2=0.1 which is reported for brain (section 13.1).

5.7 Discussion

Unlike general purpose implementations of the Gauss Newton method, we are facing additional speed requirements in order to reach satisfactory performance on standard PCs for voxel based evaluation, given a quarter million of brain voxels in the images that we had, and a possible higher number in any “high resolution” applications the program might still be used for[2]. At the same time, we enjoy the benefit of low dimensionality and a simple and rather uniform structure of the Model Function $\mathcal{F}$, whose definition depends on C_t and C_r. This, however, is compromised by noise of these curves, especially of C_t.

Speed requirements were met by limiting the number of iterations to 20, yielding performance of 5000 voxels per second on a 2.67 GHz single processor machine. We thus lost 7.55 percent of voxels whose convergence was too slow for the deadline, but regained 5.73 percent by relaxing the break-off threshold after iteration 20, at a moderate price paid in terms of precision.

Depending on the subject, between 5 to 15 percent of all voxels fail to evaluate completely. Of the voxels that do converge, a fraction of about 1

[2] although it defies the limitations of PET resolution (section 1.1.4), smaller voxel sizes are delivered by certain PET scanners, or might be employed in future studies as part of an effort to improve on atlases or templates.

per mille was found where S_1=(1.;0.1;0.1) was not in the attractor range of the global optimum. They had competing optima whose k_2 was either much lower or higher than 0.1 which is assumed realistic for most of the brain. A closer look at the corresponding TACs revealed that some were very noisy (high $noise_{rel}$ as of section 4.7.2, low overall intensity), others resembled the reference curve (high *simcr* as of section 4.6.3).

We conclude that the vast majority of voxels has sufficiently stable convergence properties to render the method applicable from S_1 as a starting point. Others are less suitable for kinetic analysis and mostly cause the method to fail. A tiny fraction of these converges under the chosen regime, they exhibit two local optima located far from each other, and it depends on chance and the starting point which one of these optima is found.

Chapter 6

Noiseless Validation

Given a triple of kinetic constants k_1, k_2, k_3, we can simulate perfect noiseless data by plugging them in the model equations. We then expect NLS and COLOGNE, as a minimum requirement, to be able to retrieve the known kinetic constants.

As we gradually move the simulation closer to reality, there emerge various sources of unavoidable bias: interpolation, discretization, failure of Assumption 4. Their effect will be quantified in this chapter. It provides reference data for re-implementers of the method, and a base for the following chapter, where noise is included in the simulation.

6.1 Generating synthetic Input

Target and reference TACs will be generated from equation (3.4), which depends on a Blood Input Function C_{pl}. In real data, the Blood Input Function is different for every patient. For the simulations, we need to make a fixed choice for C_{pl} and the framing schedule.

6.1.1 Framing Schedules

Let N be the number of frames. t_k (k=1,...,N) is the end time of frame k and the beginning of frame k+1, in minutes after tracer injection. t_0, the beginning of the first frame, is always set to 0.

6.1.1.1 Schedules from Literature

The framing schedule used in [14] and throughout this work unless otherwise indicated, will be called the **"traditional"** one. It has been used for the subjects of Samples 1 through 4 (section 2.11). It is listed in Table 6.1 next to two other schedules. The table also shows the Decay Weights of equation (2.7). Their columns add up to 0.869798, which is the fraction of total nuclide that decays during the first 60 minutes.

	“Traditional”		Iso20		Iso60	
	end time	decay-	end time	decay-	end time	decay-
Frame nr.	[min]	weight	[min]	weight	[min]	weight
1	0.5	0.01685	1.31	0.04349	0.43	0.01450
2	1.0	0.01656	2.68	0.04349	0.87	0.01450
3	1.5	0.01628	4.11	0.04349	1.31	0.01450
4	2.0	0.01601	5.62	0.04349	1.76	0.01450
5	2.5	0.01574	7.22	0.04349	2.21	0.01450
6	3.0	0.01547	8.90	0.04349	2.68	0.01450
7	4.0	0.03017	10.68	0.04349	3.15	0.01450
8	5.0	0.02916	12.58	0.04349	3.63	0.01450
9	7.5	0.06871	14.62	0.04349	4.11	0.01450
10	10	0.06312	16.80	0.04349	4.61	0.01450
11	15	0.11123	19.15	0.04349	5.11	0.01450
12	20	0.09385	21.72	0.04349	5.62	0.01450
13	25	0.07919	24.52	0.04349	6.15	0.01450
14	30	0.06682	27.63	0.04349	6.68	0.01450
15	35	0.05638	31.10	0.04349	7.22	0.01450
16	40	0.04757	35.03	0.04349	7.77	0.01450
17	45	0.04014	39.57	0.04349	8.33	0.01450
18	50	0.03387	44.94	0.04349	8.90	0.01450
19	55	0.02857	51.52	0.04349	9.48	0.01450
20	60	0.02411	60	0.04349	10.08	0.01450
...					...	...
58					54.08	0.01450
59					56.89	0.01450
60					60	0.01450

Table 6.1: *Framing Schedules*

In all schedules, the frames increase in length, compensating for the loss of radioactivity. The two schemes on the right have 20 and 60 frames, they are distinguished from “traditional” in that they distribute radioactive decays evenly over the frames. They are interesting since their Decay Weighting is trivial, and will be introduced in the following section.

6.1.1.2 Iso-Decay Schedules

Given the half-life τ of some isotope, a number N of frames, and the end time t_N of the last frame, we want to compute the (uniquely determined) schedule with t_0=0 that distributes radioactive decays evenly over the frames.

Equate the total amount of radioactivity injected with 1. Then by the law of radioactive decay, at time t we will have the amount

$$2^{-\frac{t}{\tau}} \tag{6.1}$$

left. At the end of frame N, this will be $2^{-\frac{t_N}{\tau}}$, and at the "end of frame 0" (which is the start of frame 1) it will be 1. Consider the function f assigning to every frame number the radioactivity left at the end of that frame, thus

$$f(k) := 2^{-\frac{t_k}{\tau}} \tag{6.2}$$

It satisfies f(0)=1 and f(N)=$2^{-\frac{t_N}{\tau}}$. Since the decrease in radioactivity shall be the same in each timeframe, we need f() to progress along a straight line connecting these two points. So we have

$$f(k) = \frac{N-k}{N} \cdot 1 + \frac{k}{N} \cdot 2^{-\frac{t_N}{\tau}}$$

It is easily seen that this indeed is a linear[1] function of k and passes through the given points. Replacing the left hand side using (6.2), we obtain

$$2^{-\frac{t_k}{\tau}} = \frac{N-k}{N} + \frac{k}{N} \cdot 2^{-\frac{t_N}{\tau}} \tag{6.3}$$

and solve this for t_k:

$$-\frac{t_k}{\tau} = \log_2 \left(\frac{N-k}{N} + \frac{k}{N} \cdot 2^{-\frac{t_N}{\tau}} \right)$$

$$t_k = -\tau \cdot \frac{\ln \left(\frac{N-k}{N} + \frac{k}{N} \cdot 2^{-\frac{t_N}{\tau}} \right)}{\ln 2} \tag{6.4}$$

By this formula, we compute the frame end times. As required, it returns t_0=0 and t_N=t_N.

By **Iso20**, we denote the framing schedule arising from equation (6.4) with N=20 and t_N=60. Iso20 and Iso60 will be used in simulations to have a comparison with the "traditional" scheme.

6.1.2 Blood Input Functions

Tracer in blood plasma is quickly degraded, by both hydrolysis and uptake in tissue. The simplest way of modelling this decline is a mono-exponential function:

$$C_{pl}(t) = H \cdot \exp(-k_p \cdot t) \tag{6.5}$$

[1] more precisely: affine

where H and k_p are new constants. It has the advantage that C_t and C_r of equations (3.4) and (3.5) can be expressed by explicit formulae. By trivial calculation, one gets

$$C_t(t) = -\frac{k_1 \cdot k_2}{k \cdot (k - k_p)} \cdot H \cdot e^{-kt} \cdot \left(1 - e^{(k-k_p)t}\right) + \frac{k_1 \cdot k_3}{k \cdot k_p} \cdot H \cdot \left(1 - e^{-k_p t}\right) \quad (6.6)$$

$$C_r(t) = H \cdot \frac{k_{1r}}{k_p} \cdot \left(1 - e^{-k_p t}\right) \quad (6.7)$$

Likewise, the model gives an explicit formula for w_k. Using $C_r = k_{1r} \cdot C_i$ (3.5) and substituting everything into either (3.14) or (3.16) leads to

$$w_k(t) \;=\; 1 \;-\; \frac{e^{-kt} - 1 + \frac{k}{k_p - k} \cdot \left(e^{(k-k_p)t} - 1\right)}{1 - e^{-k_p t}} \quad (6.8)$$

6.1.2.1 Two Standard Input Functions

It remains to determine H and k_p for equation (6.5). In [36], there is a diagram (Fig. 2) showing the time course of unhydrolyzed tracer in the arterial blood of a single subject. The curve consists of an initial peek smoothing out to a monoexponential tail (that appears linear on the logarithmic plot). From that diagram, we extracted the parameters of the monoexponential part:

$$H = 0.2$$
$$k_p = 0.1197$$

To obtain a more realistic model of C_{pl}, a biexponential function is better suited. We therefore created the **S3 Input Function**, a biexponential fit to the average of the input curves found in Sample 3. Section 7.1.1 describes in detail how this function was obtained. Its equation is

$$C_{pl}(t) = 0.213669 \cdot \exp(-1.096555 \cdot t) + 0.0078699 \cdot \exp(-0.127378 \cdot t) \quad (6.9)$$

C_t and C_r for such biexponential Input Functions can be computed by adding the terms (6.6) and (6.7) of their monoexponential summands; this is guaranteed to work by Theorems 4.1 and 4.2.

6.2 Linearity Tests

The following tests helped us through the early stages of implementing COLOGNE and may be appreciated by re-implementers of the method.

6.2.1 COLOGNE without Discretization

Consider again equation (3.13):

$$\frac{C_t(t)}{C_r(t)} = A \cdot w_k(t) + B \tag{6.10}$$

It asserts that, if w_k is computed for the same $k = k_2 + k_3$ as C_t and C_r, then C_t/C_r is a linear function of w_k. C_t, C_r and w_k can be computed from equations (6.6) to (6.8) once k_1, k_2, k_3, k_p and H have been chosen. Plotting the left hand side against the right hand side should result in a perfectly straight line. If the k for w_k was chosen larger than k_2+k_3, it results in a convex, if smaller, in a concave curve (see also Figure 3.1).

6.2.2 Correlation Test

This is almost the same in disguise, but it makes the peaks visible from which COLOGNE reads the maximum k. Choosing an arbitrary set of discrete times and evaluating both sides of equation (6.10) for each time, gives a cloud of dots from which we compute the correlation coefficient. While on the left side k must equal $k_2 + k_3$, we use arbitrary k for w_k on the right side. We thus have a function mapping k to the correlation. It should have a peak at k=$k_2 + k_3$. As we are dealing with noise-free model functions, the peak correlation must equal 1.

6.3 Interpolation Bias

The previous tests are only good for catching computation and programming errors. We now discuss the first real source of bias, interpolation.

w_k as required for the COLOGNE method (equation 3.16) and $\mathcal{F}$ as required for NLS (equations 4.10 and 4.11) both depend on integrals of C_r, hence C_r must be interpolated. Its interpolated version is a linear or cubic spline, hence it cannot match formulae (6.6) and (6.7) which are used for the simulations. In the terminology of section 4.2, the latter are not fully "Model Compliant". This leads to biased k_3 results, and we are now quantifying this bias.

Value tables for C_t and C_r were generated for a given set of kinetic constants, from both standard Input Functions (section 6.1.2.1) and the "traditional" framing. Then they were evaluated by NLS and COLOGNE. Two interpolation methods were tested: linear, and by natural cubic spline. **Table 6.2** shows relative errors of resulting k_3. Comparing columns 2 and 5 to 3 and 6, it is obvious that the cubic spline brings better results in most cases. This is not surprising since C_r is concave (see section 3.4.1). But even with linear interpolation, the bias is moderate. Although NLS and

	monoexponential			biexponential		
	COLOGNE		NLS	COLOGNE		NLS
k_3	linear	spline	spline	linear	spline	spline
0.01	1.6	0.06	-0.01	0.3	0.10	-0.01
0.02	1.4	0.02	-0.01	0.3	0.04	-0.02
0.05	1.4	0.02	-0.01	0.3	0.04	-0.02
0.1	1.6	0.01	-0.01	0.4	0.06	0.00
0.2	1.8	0.03	-0.01	0.4	0.18	0.05
0.3	1.9	0.03	0.00	0.4	0.32	0.12
0.4	1.9	0.07	0.00	0.5	0.50	0.21
1.0	1.6	0.22	0.07	1.4	2.28	1.21

Table 6.2: *Interpolation Bias for linear and spline interpolation, both algorithms and both standard Input Functions, in percent of the true k_3. k_2 was 0.1 in all simulations.*

COLOGNE use the same interpolation, NLS has better results in both parts of the table.

Generally, Interpolation Bias is negligible compared to other sources of bias investigated in the following. This remains the case even if k_2 is set to much lower values, which are notoriously problematic in the presence of noise.

6.4 k_{3r}-induced Bias

Both COLOGNE and NLS assume that the reference region is ideal, so C_r relates to the Blood Input Function C_{pl} via equation (3.5). In reality, it is more like equation (3.4) with k_{3r} and k_{2r} of the reference region plugged in for k_3 and k_2. The difference leads to **"k_{3r}-induced bias"** in the final k_3 results. k_{3r}-correction (section 3.2.3) can be used to avoid this type of bias, provided k_{3r} and k_{2r} are known.

The bias was quantified by the following experiment. For 6 combinations of k_{2r} and k_{3r}, reference curves were simulated. Each was used with and without appropriate k_{3r}-correction, bringing the total number to 12. They were all interpolated by cubic splines. TACs were simulated for k_2=0.1 and different values of k_3, and evaluated with NLS against each reference curve. The S3 Input Function (section 6.1.2.1) has been used for all simulated C_r and C_t. **Table 6.3** shows relative errors of the resulting k_3 in percent.

The bias is almost always negative. For Putamen with k_{3r}=4 and $k_3 <$ 0.3 it is negligible (column 2). For high k_3 it picks up, which is not surprising as then the TACs start resembling the reference curve, so errors of the latter have a larger impact. A similar effect (not listed in the table) was observed for low k_2.

	$k_{2r} = 0.1$			$k_{2r} = 0.3$		
k_3	$k_{3r} = 4$	$k_{3r} = 2$	$k_{3r} = 1$	$k_{3r} = 4$	$k_{3r} = 2$	$k_{3r} = 1$
0.01	-0.44	-1.72	-6.19	-1.23	-4.71	-16.01
	-0.01	-0.02	-0.05	-0.03	-0.06	-0.13
0.02	-0.33	-1.23	-4.33	-0.90	-3.35	-11.14
	-0.02	-0.03	-0.05	-0.04	-0.06	-0.11
0.05	-0.31	-1.15	-3.92	-0.85	-3.13	-10.04
	-0.02	-0.03	-0.05	-0.03	-0.06	-0.11
0.1	-0.42	-1.59	-5.20	-1.19	-4.35	-13.35
	-0.01	-0.02	-0.04	-0.02	-0.06	-0.13
0.2	-0.78	-2.98	-8.94	-2.32	-8.17	-23.52
	+0.04	+0.01	-0.03	+0.0	-0.08	-0.21
0.3	-1.27	-4.69	-13.02	-3.83	-12.96	-36.06
	+0.09	+0.05	-0.02	+0.02	-0.12	-0.34
0.4	-1.87	-6.60	-17.07	-5.67	-18.43	-51.22
	+0.16	+0.08	-0.01	+0.03	-0.20	-0.51

Table 6.3: *k_{3r}-induced Bias. Relative errors of k_3 in percent, when TACs are evaluated against nonideal reference curves, without (upper figure in every block) and with (lower figure) appropriate k_{3r}-correction.*

As columns 3 and 4 are showing, the bias gets larger if k_{3r} is lower. It approximately triples when k_{2r} is raised from 0.1 to 0.3. Throughout the table, k_{3r}-correction is working nicely, improving precision by one or two orders of magnitude. However, knowing the right k_{2r} and k_{3r} is critical. For instance, consider $k_{2r} = 0.1$, $k_{3r} = 4.0$ (as assumed in Putamen, section 13.1) with $k_2 = 0.1$, $k_3 = 0.2$. Without correction, the bias is -0.78%. It goes down to +0.04% if the right correction is applied. Correction with k_{3r} below 4, or k_{2r} higher then 0.1, will invert and eventually increase the bias. It becomes +0.65%, +2.25%, +7.91% if the correction is performed with k_{3r}= 3, 2, 1 respectively.

The analogous situation for the combination $k_{2r} = 0.1$, $k_{3r} = 2.0$[2] is shown in **Table 6.4**. Without correction, we expect -2.98% of bias. There

[2] this might be more realistic for Putamen reference curves, considering possible PVE effects

k_{3r} used for correction	(none)	k_{2r} used for correction 0.1	0.2	0.3	0.4
(none)	**-2.98**				
4		-2.17	-1.40	-0.66	+0.05
3		-1.57	-0.23	+1.05	+2.28
2		**+0.01**	+2.81	+5.43	+7.90
1.5		+1.89	+6.37	+10.50	+14.33
1		+5.70	+13.19	+19.75	+25.55

Table 6.4: *k_{3r}-induced bias after miscorrection. C_r is simulated for k_{3r}=2 and k_{2r}=0.1, and then "corrected" using different combinations of k_{3r} and k_{2r}. Showing relative error of k_3 in percent, after evaluating C_t for $k_2 = 0.1, k_3 = 0.2$ against the resulting C_r. Bold print: without correction, with appropriate correction.*

are 11 combinations of k_{2r} and k_{3r} that lead to smaller bias. As we are farther away from the ideal situation, the general level of bias is higher than in the previous example.

6.5 Discretization Bias

There is a fundamental problem in designing a continuous kinetic model and expecting dynamic PET data to conform to it. The model assigns activity to every point in time, but the PET scanner can only measure activity averaged over time intervals. So it delivers a value table consisting of weighted integrals over the Model Curves. Therefore, TACs from real PET data will not exactly match equations (3.4) and (3.5), let alone equation (4.11). So we collect activities from intervals and map them to time points. Questions are:

- Which is the smartest way to perform this mapping, and
- how much bias is to be expected?

It seems obvious that the time points should be close to the frame centers. All above tables were computed with this policy. Mapping time to the frame ends was found to produce some 20 percent of negative bias with the "traditional" framing (data unlisted). One might argue that time points belong a little left of the center, since the Decay Function is declining, so there is more radioactivity toward the beginning of each frame. We started experimenting with such an approach, mapping them to the centroids of the areas under the Decay Function, but reached only slight improvement. This is not so surprising since there is probably *no* approach that leads to Model Compliance in the sense of the definition of $\mathcal{F}$ (equation 4.11).

	k_2					
k_3	0.01	0.02	0.05	0.1	0.2	0.5
0.01	-1.7	-1.8	-1.9	-1.9	-1.0	+0.7
0.02	-1.2	-1.3	-1.4	-1.5	-1.0	+0.0
0.05	-1.1	-1.2	-1.3	-1.3	-1.1	-0.4
0.1	-1.4	-1.4	-1.4	-1.4	-1.1	-0.5
0.2	-1.7	-1.7	-1.6	-1.5	-1.2	-0.6
0.3	-1.7	-1.7	-1.6	-1.5	-1.3	-0.7
0.4	-1.8	-1.7	-1.7	-1.5	-1.3	-0.7
1.0	-1.6	-1.5	-1.5	-1.4	-1.3	-1.1

Table 6.5: *Discretization Bias of the "traditional" framing. Relative errors of k_3 in percent.*

6.5.1 Simulating PET Data by Integration

In order to assess the resulting bias, we need realistic TACs. Formulae (6.6) and (6.7) are based on time points. They need to be modified by

1. performing reverse Decay Correction in order to get "radioactive" functions, which are proportional to decays per second

2. integrating these functions between the frame borders.

The modified formulae will be used for the following section, and throughout chapter 7. While they are too lengthy to be presented here, they are still easy to implement. Given one of the standard curves of section 6.1.2.1 as input, formulae (6.6) and (6.7) are sums of constant and exponential terms. This remains the case after reverse Decay Correction, which is a multiplication with $\exp(-ln(2) \cdot \frac{t}{\tau})$. The integrals of the resulting function are still of this type, so everything can be expressed in closed terms.

6.5.2 Measuring the Bias

TACs were simulated as just described, for an array of k_2/k_3-combinations, using the S3 Input Function and the "traditional" framing. Combined Correction as performed by the reconstruction software (section 2.6.1) was applied. They were then evaluated by NLS against ideal reference curves, mapping time to the frame centers.

Table 6.5 has relative errors of k_3. We find considerable increase over the background of Interpolation Bias. There is little variation throughout the table with the exception of its right upper corner. This type of bias strongly depends on the framing schedule. With Iso20 and Iso60, it almost disappeared.

Chapter 7

Monte Carlo Simulations

In this chapter, we investigate the behavior of both algorithms when faced with noisy input. We start with a Model Curve and add specific amounts of noise to every frame to make simulated data. The advantages of this approach are:

- we know what the right result should be
- by using different sets of random numbers, we obtain samples of k_3 and can study their empiric standard deviations.

7.1 The Simulator

Figure 7.1 shows a generalized and simplified view of the simulator workflow, both for simulating C_t and C_r. We begin by setting the kinetic constants k_1, k_2, k_3 (or their counterparts k_{1r}, k_{2r}, k_{3r} of the reference region) to fixed values. Then, from the model equations, the framing and the isotope half life, we compute the distribution of the decays to the frames (called "noiseless curves"). Now there is a fork with two branches.

In the main branch, we apply Combined Correction (section 2.6.1) like the reconstruction software, leading to almost Model Compliant[1] curves. Then every frame receives its own amount of simulated Gaussian noise, leading to a (target- or reference) TAC that comes close to real image data. It is then evaluated like the real data, retrieving the kinetic constants that were provided as input, with both their bias and (by repeating the simulation often) their empiric SD.

In the **Noise Assessment Branch** on the right side of the diagram, we decide how much noise must be simulated for every frame. This depends on the number of decays taking place during the frame in a fixed volume, and the associated Poisson distribution. It also depends on how this noise

[1] Section 6.5 tells why they can't be fully Model Compliant.

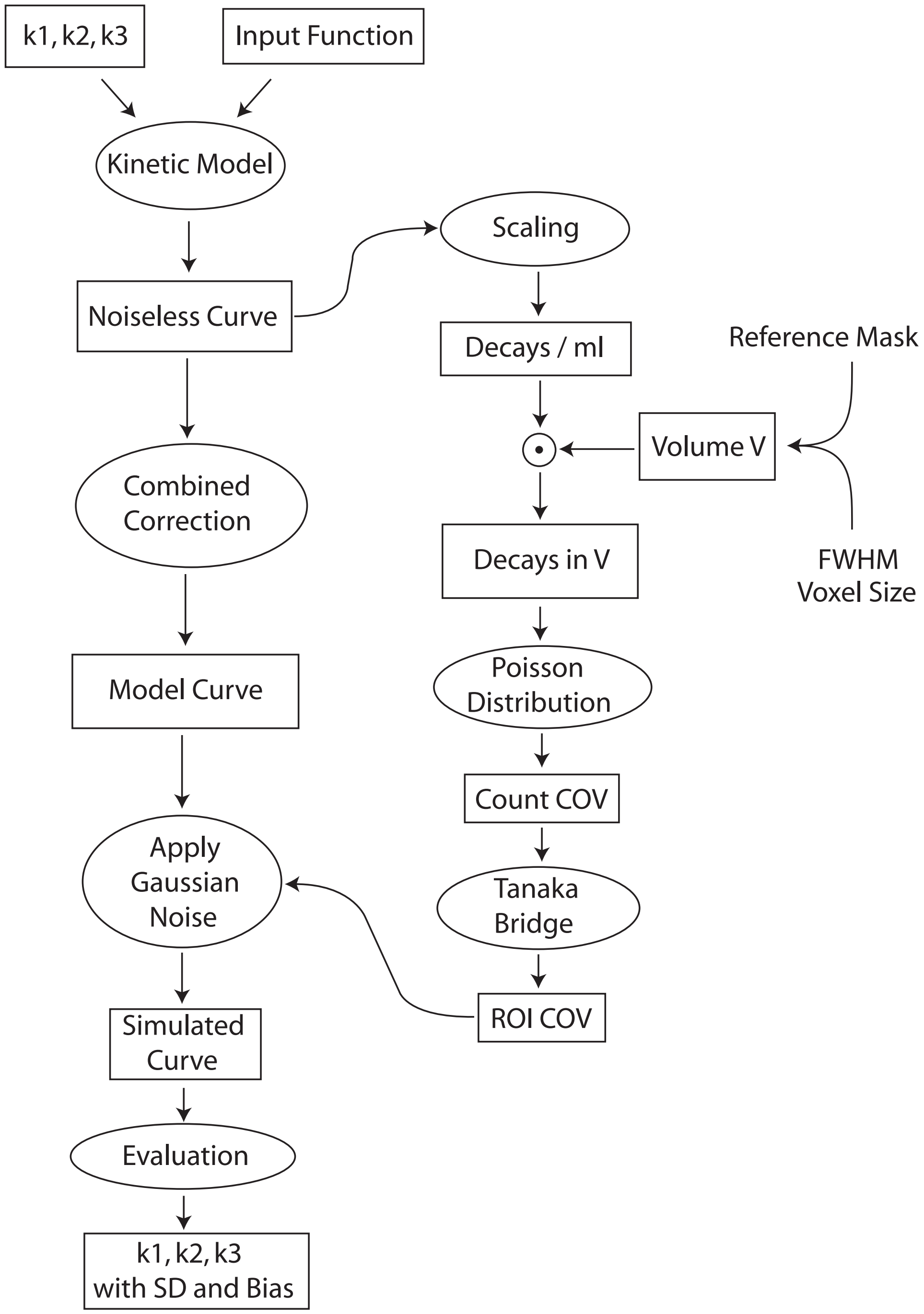


Figure 7.1: *The Simulator Workflow*

propagates through the reconstruction process, and how the images are preprocessed before sampling the TACs. So this assessment is complicated and involves a number of assumptions.

This section explains the details of the diagram, and how the simulator was calibrated to match conditions of the scanner and software setup in Cologne.

7.1.1 The Blood Input Function

In real PET data, this function will be different from patient to patient, its shape depending on the dynamics of injecting the tracer, the properties of the subject's blood circulation system and his metabolism. Its amplitude depends on the amount of tracer injected. The amplitude is no matter of concern, since by Theorem 4.5 it does not influence k_3.

For the simulator, we need a standardized Input Function that is somehow representative of the real situation. That **"S3 Input Function"** was obtained by a biexponential fit to the patients of Sample 3. Its equation

$$C_{pl}(t) = 0.213669 \cdot \exp(-1.096555 \cdot t) + 0.0078699 \cdot \exp(-0.127378 \cdot t)$$

was obtained by the following procedure. The subjects' reference curves were sampled with a 674+2 Putamen Mask (see section 10.3.1). They were scaled to similar amplitude (by having the sum of their squares equal 1) and then averaged across the subjects. To the resulting value table, a function

$$f(t) = H_0 - H_1 \cdot e^{-k_1 \cdot t} - H_2 \cdot e^{-k_2 \cdot t} \tag{7.1}$$

was fitted, by choosing H_0, H_1, H_2, k_1 and k_2 such as to minimize the squared differences between the averaged measurements and the values of f. The S3 Input Function is the derivative of f.

7.1.2 Noiseless Curves

are generated by the integration method of section 6.5.1. They come in units proportional to counts per timeframe and milliliter. To convert to specific activity, we apply Combined Correction as in section 2.6.1. In the Noise Assessment Branch, by contrast, we need true decays[2] per milliliter. This requires suitable scaling.

7.1.3 Scaling toward true Decays

To find that scaling factor, it would help if the decays per milliliter were known for one region. Suppose, for instance, we knew which fraction of the injected tracer ultimately decays in the brain. We then could compute from

[2] not to be confounded with "counts", which refers to decays detected by the scanner

the injected dose the overall number of decays that take place in the brain, and by additional assumptions, might obtain similar information for the area of interest. But while the brain fraction is easy to obtain (see section 7.1.3.1), the “additional assumptions” tend to be speculative. In fact, they made us drop the whole-brain based approach.

Instead, we focus on the reference region and try to obtain the fraction of total injected nuclide that decays in 1 milliliter (ml) of it. This will be called **Putamen Decay Fraction** or **Cerebellum Decay Fraction**. Together with the injected dose, it allows to calculate the number of decays per ml of Putamen or Cerebellum in infinite time. To obtain the scaling factor, this number is divided by its counterpart of the simulated noiseless reference curve. This is the sum of all frames, plus an estimate extrapolating it to infinite time.

We thus obtain a scaling factor for the reference curve C_r. This factor is used again in the simulation of C_t. This is legitimate since the amplitudes of C_r and C_t are in a fixed relation to each other, as determined by the kinetic model. That relation must still be the same after scaling.

7.1.3.1 Measuring the Decay Fractions

The Decay Fractions (for Putamen and Cerebellum) are determined by averaging over a number of measurements from real images. The reason why they are based on 1 ml and not on the whole Putamen or Cerebellum, is that it allows to use the reference volume as another free parameter in the simulations.

The total number of decays can be obtained from reconstructed images, which are in units of Becquerel per milliliter, as follows. Take the intensity of each frame times its length to obtain decays per ml, add them up over the frames and add an “infinity frame” to account for all atoms decaying later. The decays per ml of the infinity frame are computed from the intensity of the constant tail of the TAC (computed as a suitable mean of the latest frames) times its Decay Weight by formula (2.7).

Thus from the normalized frames we compute an **“Infinity Image”**. It is almost the same as the Decay Weighted sum image and different only by inclusion of the infinity frame, and by being in units of decays per ml.

The Infinity Image can then be sampled in order to obtain averages, for certain regions or for the whole brain. Sampling the reference mask will give the number of decays per ml of the reference region. To obtain aforementioned Decay Fraction, it is divided by the total number of injected ^{11}C atoms, which is computed from the injected dose by equation (2.1).

For the whole-brain based approach, we need the Decay Fraction for the whole brain. While this is no longer required in the current version of the simulator, it is still an interesting figure to have. The Infinity Image is sampled for the whole brain, giving decays per ml in infinite time. They are then taken times the subject's brain volume, and divided by the total number of injected ^{11}C atoms.

7.1.3.2 Brain Volume

The brain volume was obtained from the stretch factors (one for each axis) that were recorded by VINCI during the normalization step. On the normalized image, the voxels are sized 2 times 2 times 2 millimeters, and their number (under the full brain mask of section 10.1) was 251461, resulting in a volume of 2011.688 ml. Dividing this by the product of the stretch factors, gives the brain volume on the native image.

7.1.3.3 Results

By applying the above procedures to the PET data of Sample 1, the following averages and standard deviations were obtained:

- Brain volume: 1463 ± 141 ml
- Decay Fraction for 1 ml of Putamen: 0.0000672 ± 0.0000269
- Decay Fraction for 1 ml of Cerebellum: 0.0000597 ± 0.0000247
- Decay Fraction for the whole brain: 0.03801 ± 0.01369

Raw data of this computation, and corresponding results of patient samples from Milan, are in appendix (A.5).

7.1.4 The Tanaka Bridge

Next, we need to assess noise propagation through the reconstruction process. Consider a fixed volume **V**. Since we know the total decays per ml, we also have them for V. The actual number is Poisson distributed, where the variance equals the expectation. From this we obtain the variance, SD, and coefficient of variation (COV) of the original shot noise.

Now the challenge is to transform this to an estimate for the noise in the reconstructed image. For this purpose, Tanaka et al. [36] built an empiric bridge, linking the above COV (to be called **Count COV**) to the COV of image intensity in the volume V, to be called **ROI COV**. They reached the formula

$$ROI\ COV = 29.1 \cdot CountCOV + 0.009 \tag{7.2}$$

for their specific scanner and software setup. It was obtained by measuring Count COV and ROI COV for 2 cortical ROIs of 16 time frames of 3 healthy human subjects. That gives a total number of 96 data points, and the above equation is the regression line through that cloud of points.

7.1.4.1 Critizising the Tanaka Bridge

The question is if we may use the Tanaka formula as it is, for calibrating the simulator. For 3 reasons, we decided against it:

1. Error propagation depends on the scanner and the reconstruction software, and any filters used therein. The Tanaka formula needs to be adjusted to every new setup.

2. In the Tanaka paper, the following formula is used to compute ROI COV of each data point:

$$ROI\ COV = \frac{Pixel\ COV}{\sqrt{N = number\ of\ pixels\ in\ ROI}} \tag{7.3}$$

   where "Pixel COV" is the coefficient of variation of the pixel intensities in the ROI. The formula is valid for stochastically independent voxels. Are the voxels of a reconstructed image stochastically independent? Generally, no, but they might come "close" to being so. The degree to which they are, depends on the reconstruction method and all steps of image preprocessing up to the point where TACs are sampled. This includes normalization and smoothing.

3. It appeared like "CountCOV" in the Tanaka paper refers to true decays, not detected decays, but this was not explicitely stated, and the name provided for additional confusion. The difference between the two interpretations is huge, since scanner sensitivity is in the region of 3 percent.

7.1.4.2 Phantom Experiments

In addition to repeating the Tanaka procedure, we decided to assess ROI COV directly, as the empiric COV of a sample of ROIs. Samples of truly homogenous ROIs are easiest obtained from phantom data. Note that homogeneity is a must if equation (7.3) is used, since any variation of intensity inside a ROI increases the enumerator.

The phantom was a cylinder of 14.2 cm diameter, containing two smaller cylinders of 4.1 cm diameter each, whose volumes were 162 and 84 ml, leaving 1900 ml for the big cylinder. All cylinders were filled with water and injected with ^{11}C tracer, to reach initial activities of 12300 Bq/ml in the large and the smallest cylinder and 56300 Bq/ml in the medium cylinder. Geometry and activities were intended to represent the closest match attainable, to a human head containing hot Basal Ganglia and less active cortex regions, although for the latter, 25000 Bq/ml would have been closer to reality. For measurement of noise, it is important to match everything to the real situation, as the noise also depends on the scatter fraction and the background of random events, which are estimated and subtracted by the reconstruction software. Acquisition was carried out in 30 frames of 5 minutes.

On the reconstructed images, homogenous regions of the large and medium cylinder were isolated and dissected into mutually disjoint and roughly isoperimetric ROIs. To the big cylinder, 3 dissections were applied, where the ROIs had 100, 1000 and 10000 voxels apiece, sized 2x2x2 mm. The number of ROIs per dissection was 1527, 152 and 15. The small cylinder was dissected twice, into 15 ROIs of 674 voxels, which is the size of the standard Putamen mask (section 10.3.1), and 104 ROIs of 100 voxels.

From the initial activity, the frame times, ^{11}C half life and the ROI volume, the expected number of decays was computed, and, from it, Count COV (section 7.1.4). ROI COV was computed twice: first, following Tanaka, by equation (7.3), second, from the empiric SD of the average intensities of all ROIs of the dissection. This was performed for each of the 30 frames. Instead of linear regression, we looked at

$$\frac{ROI\ COV}{CountCOV} \tag{7.4}$$

of every frame, calling them **"Tanaka quotient"** and **"empiric quotient"** depending if ROI COV was computed by equation (7.3) or not.

7.1.4.3 Results

In the large cylinder, the Tanaka quotients were all close to 32, except for the early frames, where they were somewhat higher with a maximum around 36. This was observed on all 3 dissections. The higher readings in the early frames may be attributable to a loss of scanner sensitivity due to increased dead times and, possibly, random and scatter effects. On the medium cylinder all Tanaka quotients were close to 20, again with higher readings on the early frames.

In most of the frames, the empiric quotients were roughly twice as large as the Tanaka quotients, 60 for the large and 40 for the small cylinder. On the 674 voxel dissection, they averaged 30, the ROIs of this dissection were not isoperimetric but "Putamen shaped". In the early frames, empiric quotients were higher, in particular on the 1000 and 10000 voxel dissection of the large cylinder, where values of up to 300 were seen.

7.1.4.4 Discussion

- As 32 is close to the factor 29.1 of equation (7.2), it confirms the Tanaka result and assures that "CountCOV" refers to decays rather than counts.

- Finding the empiric quotients twice as large as their Tanaka counterparts is in accord with our expectation that equation (7.3), as a result of assuming voxel independence, underestimates ROI COV.

- Much higher empiric quotients were seen in the early frames. Our explanation is that enumerator and denominator of equation (7.4) are small in high activity frames. Therefore, any systematic effect that increases the enumerator weighs in heavily on the result. Such effects could be poor mixing of the tracer with the water, or local differences in scanner sensitivity.

- It is not understood why the Tanaka quotients could be so different in the small and large cylinder. Attenuation effects would explain only differences of up to 10 percent.

It was decided to trust the empiric quotients of the late frames in the big cylinder and replace the Tanaka formula with

$$ROI\ COV = 60 \cdot CountCOV \tag{7.5}$$

7.1.4.5 Conclusion

The data are certainly not of the quality that justifies performing linear regression and including its constant term in the final result. The factor 60 appears more trustworthy since it was seen in all 3 dissections and formed a long constant tail in the 100 voxel dissection. However, there remains a speculative element in the interpretation. The Tanaka Bridge therefore is the weakest component of simulator calibration. At least, the meaning of "Count COV" could be clarified.

7.1.5 Equivalent Volume

Now the decays per milliliter are known after suitable scaling (section 7.1.3), and we need to consider a volume **V** to reach the near end of the Tanaka Bridge. For the reference curve C_r, V is the volume of the reference mask. For voxel based TACs C_t, we must take account of the Gauss filtering step, which is part of image preprocessing and introduces a high level of stochastic dependence between the voxels. The way how we factor this in is by choice of V.

Thus for every combination of a voxel volume **v** and a Gauss Filter width FWHM, we compute an equivalent volume V, such that the COV of a signal coming from that volume of an unfiltered image, is the same as the COV of the signal coming from one voxel of the filtered image.

Consider a voxel of volume **v** and assume that it is surrounded by a grid of like voxels. Let **n** denote the expectancy of counts in each voxel. By the

Poisson distribution, n is also its variance, therefore, $\sqrt{n}$ is its SD and $\frac{1}{\sqrt{n}}$ is its COV.

By the same argument, we get a COV of

$$\frac{1}{\sqrt{kn}} \tag{7.6}$$

for a region whose size is k times that of a voxel. Now by Gauss filtering, the voxel gets assigned a linear combination of the intensities of itself and a number of neighboring voxels. This gives the expectancy

$$E = \sum_{i=1}^{N} \lambda_i \cdot n \tag{7.7}$$

for its number of counts after filtering, where N is the number of contributing voxels and the λ_i are the coefficients of the filter kernel. Since the sum of such λ_i should be 1, we end up with E=n.

If the contributions of all voxels are independent random variables[3], their variances add up to the variance of the filtered voxel. The individual variances are

$$\sigma_i^2 = n \cdot \lambda_i^2$$

and their sum is

$$\sigma^2 = n \cdot \sum_{i=1}^{N} \lambda_i^2 \tag{7.8}$$

Taking the square root and dividing by the expectancy E=n, gives

$$COV = \sqrt{\frac{\sum_{i=1}^{N} \lambda_i^2}{n}} \tag{7.9}$$

for the considered voxel. Now equate this with $\frac{1}{\sqrt{kn}}$ which is the COV of a k-voxels sized region. This gives $k = \frac{1}{\sum_{i=1}^{N} \lambda_i^2}$. The equivalent volume is k times the voxel volume v:

$$V = \frac{v}{\sum_{i=1}^{N} \lambda_i^2} \tag{7.10}$$

Note that the denominator is smaller than 1, since the squares of the lambdas must be less than the lambdas themselves, which add up to 1. **Table 7.1** shows the equivalent volume as a function of voxel volume and Gauss Filter size, computed for the Gauss Filter kernels used in VINCI.

[3] regarding this assumption see the following section

Voxel Volume	FWHM [mm]					
$[mm^3]$	2	5	8	10	12	15
1	27	426	1747	3411	5895	11514
8	16	426	1747	3411	5895	11514
27	28	409	1747	3411	5895	11514
125	125	248	1643	3395	5894	11514
1000	1000	1000	1167	1981	4115	10382

Table 7.1: *Equivalent volumes of VINCI Gauss Filter kernels in mm^3, depending on the voxel volume and FWHM*

One can see that, for small voxels, the equivalent volume depends almost entirely on FWHM. Raising the voxel size results in a temporary drop, then it rises again just in time to stay above the volume of one voxel, as it must according to equation (7.10).

Finding a drop in response to increasing the voxel volume seems illogical. It is due to the fact that the definition of the filter kernels themselves is not quite correct. **Table 7.2** was computed for filter kernels designed by the method of section 2.9.5. It shows more plausible behavior than Table 7.1.

Voxel volume	FWHM [mm]					
$[mm^3]$	2	5	8	10	12	15
1	32	438	1766	3435	5923	11549
8	36	475	1823	3507	6009	11656
27	43	531	1921	3628	6153	11835
125	127	557	2200	4022	6626	12419
1000	1000	1118	2392	4459	7726	14735

Table 7.2: *Equivalent volumes of alternative Gauss Filter kernels*

On assuming stochastic Independence

The path to equation (7.10) involves an assumption we refused to accept for the Tanaka Bridge: stochastic independence of voxels. Dependence can be induced by the reconstruction process, and by image resampling that occurs during normalization (section 9.1). Minor degrees of stochastic dependence have little effect when followed up by substantial Gauss filtering. To model this situation, we assume another Gauss filtering step had been applied to a primordial image of truly independent voxels. Then we can apply equation (2.13) to compute the combined effect of both steps. If the first step had an FWHM of, say, 3mm, and the second step of 8mm, it would result in 8.54 mm for the combined effect. On PET data from Cologne, smoothing during reconstruction is kept to a minimum. We therefore decided to ignore dependence effects.

7.1.6 Volume Correction

Note that C_t and C_r are sampled from the normalized frames, where the brain volume equals 2012 ml (see section 7.1.3.2). The average brain volume of the Cologne MCI subjects is 1463 ml. Every volume measured on the normalized frames corresponds to a smaller volume of the native images, the correction factor being $\frac{1463}{2012}$. That factor is therefore applied to the volume V of the noise assessment branch before computing "Decays in V".

7.1.7 Applying Gaussian Noise

The Noise Assessment Branch computes COV separately for every frame. Then the frame intensity is multiplied with a normally distributed random number of $\mu = 1$ and $\sigma = COV$. Such random numbers were generated by the polar method proposed in Knuth [22].

7.2 Simulation Types and Parameters

7.2.1 Double Random Simulation

This simulation type generates a new pair of C_r and C_t in every iteration. Scaling of C_r to decays per ml (first box of the Noise Assessment Branch) is performed so as to match the injected dose times the Putamen Decay Fraction (sections 7.1.3 and 7.1.3.3). The resulting factor is also used for C_t. Computation of "COV" and application of noise are then performed separately for every frame of C_r and C_t. The frames are collected to give a pair of simulated, noisy TACs which is then subject to kinetic analysis. We obtain k_3 which is affected by both types of noise. From many iterations, we obtain a sample of k_3 on which we do statistics, computing its mean, SD and COV.

Double Random Simulations are used in this chapter to monitor general performance of the methods and weightings, and in Chapter 8 to validate *k3var*.

7.2.2 C_r Block Simulations

are used to assess error caused by noise of C_r in isolation. Unlike error caused by noise of C_t, which has a chance to average out across a region, C_r-inflicted error strikes through to regional results. This is because all target TACs of an image are evaluated against the same reference TAC C_r.

We simulate this by running N blocks of M iterations apiece, using different C_t but the same C_r in each block. If M is sufficiently large, noise effects caused by C_t will average out in every block, while the effects of C_r persist. They are measured as the standard deviation of the N block based mean values of k_3.

Implementation detail: the first iteration of a block generates C_r, C_t and the scaling factor of section 7.1.3. In the remaining iterations, there is no C_r being generated and hence no scaling factor, so we use the factor of the first iteration.

7.2.3 Fixed C_r Simulation

is used to measure the properties of one C_r sampled from a real image. A large number of C_t is simulated and evaluated against this C_r, to obtain the mean value of k_3. It should, in principle, match the k_3 used to simulate the C_t, but usually does not,

- because the C_t have been simulated for a standardized input curve that is not represented by C_r
- because of the noise of C_r.

This simulation type allows to compare reference curves while excluding all external influences, and has been used extensively in chapter 14 and section 11.1.2. As we are only interested in the mean values, we can skip the details of noise assessment.

7.2.4 Standard Parameters for all Simulations

In each of the following sections one or two parameters are variable and the rest is kept as follows:

- Injected dose: 555 MBq
- S3 Input Function (see section 7.1.1)
- Framing: "traditional" (see section 6.1.1.1)
- Noiseless curves: generated by the integration method of section 6.5.1
- C_r: assuming an ideal reference region.
- C_t: Simulated for k_2=0.1 and k_1=0.7222. The latter is a mean value obtained from cortex of Sample 1 subjects.
- Putamen Decay Fraction: 0.0000672
- Reference volume: 5.392 ml as of the 674+2-Putamen mask (section 10.3.1).
- FWHM: 8mm, whose equivalent volume is 1.747 ml.

	COLOGNE				NLS		
k_3	unweighted	Cr	decay	Combined	unweighted	decay	Combined
0.01	21.9	15.3	16.6	13.7	14.1	13.2	**13.0**
0.02	16.7	11.2	12.0	10.2	10.8	9.8	**9.7**
0.05	17.6	11.8	11.5	9.8	10.8	9.6	**9.6**
0.1	27.9	16.9	16.3	13.5	15.2	**13.4**	13.5
0.2	54.8	36.1	38.4	27.0	30.4	**26.2**	26.8
0.3	65.0	54.9	56.2	47.2	55.9	47.3	**46.6**
0.4	66.5	62.2	61.2	**59.3**	76.4	69.5	64.5

Table 7.3: *k_3 COV [%] for both methods and different weightings. k_2=0.1 was used in all simulations. The lowest reading of every row is in bold print.*

- Volume Correction Factor: 0.72725
- Evaluation method: Decay Weighted NLS, no frames skipped.

7.3 Comparing Methods and Weightings

This section compares the NLS and COLOGNE methods and their weighting strategies with respect to noise propagation into k_3. The question of bias will be addressed in section 7.6.

Sections 2.7.1 and 2.7.3 advocate to use Decay Weighting with NLS. Similar reasons apply to the COLOGNE method, too. Yet in the study of 2003 [14], the frames were weighted proportional to the reference curve (C_r **Weighting**). The rationale may have been this: the first evaluation step is dividing C_t by C_r. Frames of low C_r have unfavorable noise propagation properties, in that the noise of C_t gets amplified. So they should be assigned low weights.

As there seems to be reason for either weighting of the COLOGNE method, we tried them both, separately and together (applying the product of both weightings, **"Combined Weighting"**). As the COLOGNE method responded positively, Combined Weighting was also tried with NLS, where it has no theoretical justification. The results are listed in **Table 7.3**. Every entry is the coefficient of variation (COV) of 10000 k_3 obtained from Double Random Simulations using the Standard Settings of section 7.2.4. The same pairs of simulated C_r and C_t were used in every column.

Observations

Among the COLOGNE results, the double weighted strategy outperforms the competition by a clear margin. It is also obvious that either weighting is better than none. Comparing C_r with Decay Weighting does not show a clear tendency.
While NLS appears less sensitive to the choice of weighting than COLOGNE,

	FWHM[mm]					
k_3	4	6	8	10	12	14
0.01	32.8	18.6	13.2	10.7	9.4	8.7
	32.2	17.4	11.5	8.5	6.8	5.7
0.02	25.5	14.1	9.8	7.8	6.7	6.1
	25.1	13.5	8.8	6.5	5.1	4.3
0.05	26.4	14.1	9.6	7.4	6.2	5.5
	26.1	13.6	8.9	6.4	5.0	4.1
0.1	40.0	20.0	13.4	10.2	8.5	7.5
	39.3	19.3	12.4	9.0	7.0	5.7
0.2	76.8	43.0	26.2	19.5	16.1	14.2
	76.4	40.9	24.1	16.9	13.0	10.6
0.3	90.9	71.9	47.3	33.5	27.5	23.9
	90.7	70.7	43.5	29.1	21.8	17.5
0.4	90.0	85.8	69.5	55.0	42.9	37.0
	88.9	86.5	66.6	47.3	33.9	26.8

Table 7.4: *k_3 COV [%] depending on k_3, FWHM and the reference volume which is 5.392 ml in the upper row of each section and 28 ml in the lower row.*

the weighted strategies are clearly superior to no weighting. There is a narrow and undecided competition between Combined Weighting and pure Decay Weighting. Comparing the double weighted COLOGNE strategy to Decay Weighted NLS, we see the latter win by a tight margin for $k_3 \leq 0.3$ and the former by a large margin for $k_3 = 0.4$.

Tests with similar results were performed for other choices of k_2 and FWHM.

7.4 Varying Input Noise

Three of the simulator parameters of section 7.2.4 are affecting the noise of C_r and C_t: FWHM (affecting C_t), the reference volume (affecting C_r), and the injected dose (affecting both).

The effect of FWHM and the reference volume is studied in **Table 7.4**. Each entry was computed from 10000 Double Random Simulations, using Decay Weighted NLS for evaluation. The reference volumes correspond to the standard Putamen- and Cerebellum masks of section 10.3.1.

The table shows the limits of precision as they apply to a dose 555 MBq and k_2=0.1. As expected, COV decreases as FWHM is raised, reducing the noise of C_t. For a decent error level, it seems advisable to have at least FWHM=8mm. Beyond 10mm, further increase brings only moderate gain in exchange for loss of resolution. The data show rapid deterioration for

Tracer Dose	FWHM[mm]					
[MBq]	4	6	8	10	12	14
277.5	92.2	64.5	40.0	29.2	23.5	20.4
555	76.8	43.0	26.2	19.5	16.1	14.2
1110	57.0	27.4	17.8	13.5	11.2	9.9
2220	36.6	18.4	12.3	9.5	7.9	7.0
4440	23.9	12.7	8.6	6.7	5.6	4.9

Table 7.5: *k_3 COV [%] depending on the tracer dose. k_3=0.2, k_2=0.1 were used for all simulations.*

high k_3, putting the upper limit of what can sensibly evaluated at k_3=0.3.

Noise of C_r accounts for the difference between the two entries of every block. Its impact is small at FWHM=8mm, and increases as FWHM is raised. This makes sense since as we reduce the noise of C_t, a larger fraction of the total noise becomes attributable to C_r. However, the real impact of noise of C_r is not appreciated with this type of simulation. It will be further investigated in section 7.7 and chapter 11.

The effect of the tracer dose is investigated in **Table 7.5**, where that quantity is doubled in every row. 555MBq (15 mCi) serves as point of departure, this dose was administered to Sample 1 subjects (section 2.11). Cutting it in half as in the first line, has relevance for the data splitting experiments of chapter 14. 10000 Double Random Simulations were performed for each table entry, using the "difficult" combination $k_2 = 0.1$, $k_3 = 0.2$.

Precision is seen to improve roughly proportional to the square root of the injected dose. This can be exploited to improve spatial resolution: diagonal comparisons in the table reveal that doubling the dose allows to switch from FWHM=8mm to 6mm while precision deteriorates just a little. Cutting the dosage in half requires switching from 8 to 10mm and still expect some deterioration of results.

7.5 Other Framings

We also ran simulations with different framing schedules. No schedule of 20 frames was found that could convincingly beat "traditional". Iso20 (section 6.1.1.2) led to slightly inferior results. Improvements could only be made by increasing the number of frames substantially, but were comparatively small themselves. With Iso60, the coefficient of variation was 89 percent of that obtained with the "traditional" framing for k_3=0.2, and 80 percent for k_3=0.3.

Interpolation splines start looking ridiculous at 60 frames, since they seek to include every point despite their apparent noisiness. So there is little to be gained by varying the framing schedule.

7.6 Noise-induced Bias

After discussing 3 types of bias that could be studied in noiseless simulations in chapter 6, we now turn to noise-induced bias - arising from neutral noise of the input data in the course of nonlinear processing. For a number of k_2/k_3 combinations we ran 10000 Double Random Simulations using standard parameters (section 7.2.4). Total bias is the relative error of the mean k_3 in percent of the true k_3. It necessarily contains Discretization Bias (section 6.5) as a result of how we simulated the data. So we subtracted the discretization component, resulting in noise-induced bias of **Table 7.6**, shown in line 3 of every cell.

It can be further subdivided, in bias induced by noise of C_t and noise of C_r. This is examined in section 15.9, where it turns out that noise of C_t induces positive bias, noise of C_r induces negative bias. What we see in

	k_2					
k_3	0.01	0.02	0.05	0.1	0.2	0.5
0.01		49.0 −1.8 +1.5	21.4 −1.9 −1.1	13.2 −1.9 −0.6	11.3 −1.0 −0.4	13.4 +0.7 −0.2
0.02	66.7 −1.2 +11.7	36.3 −1.3 −0.9	15.5 −1.4 −0.7	9.9 −1.5 −0.3	8.3 −1.0 −0.2	9.5 +0.0 +0.1
0.05	80.9 −1.1 +13.1	35.1 −1.2 +0.5	15.2 −1.3 −0.3	9.6 −1.3 −0.2	7.5 −1.1 −0.1	7.6 −0.4 +0.1
0.1		62.7 −1.4 +7.0	22.7 −1.4 +0.4	13.4 −1.4 0.0	9.4 −1.1 −0.1	8.1 −0.5 +0.1
0.2			53.5 −1.6 +4.9	26.2 −1.5 +1.1	15.9 −1.2 +0.2	11.0 −0.6 0.0
0.3				47.3 −1.5 +2.7	25.1 −1.3 +0.8	15.2 −0.7 +0.1
0.4					37.9 −1.3 +0.9	20.2 −0.7 0.0

Table 7.6: *Noise, Discretization Bias and noise-induced bias (from top to bottom in every cell) in percent of the true k_3. Combinations whose failrate exceeded 10 percent are not shown.*

Table 7.6 is the superposition. We observe that

- bias is largest for high k_3 and low k_2. These combinations also have the most noise, bordering areas that we've excluded from the table because of more than 10 percent of voxel loss to failselection.
- Noise-induced bias is more than an order of magnitude below the noise itself. For combinations of practical interest, it is small enough to be ignored.

7.7 C_r induced Bias

Double Random Simulations as of Table 7.6 fail to capture the effects of noisy C_r in real images. This is because they allow the influence of C_r to average out, which is not the case in the real situation.

Table 7.7 shows results of C_r Block Simulations (section 7.2.2). Its entries are COVs of samples of 100 k_3, each resulting from 1000 iterations with one simulated C_r.

	V=5.392 ml			V=28 ml
k_3	k_2=0.05	k_2=0.1	k_2=0.2	k_2=0.1
0.01	11.42	6.95	5.57	3.07
0.02	7.55	4.61	3.65	2.05
0.05	6.47	3.92	2.93	1.76
0.079	7.69	4.55	3.22	2.05
0.1	9.02	5.24	3.58	2.36
0.2	19.35	10.37	6.26	4.52
0.3	29.76	17.24	9.98	7.59

Table 7.7: *C_r Block Simulations (k_3 COV in %). 0.079 is the average k_3 found in cortex of subjects of Sample 1*

7.7.1 Discussion

Cr-induced bias has similar properties as the noise in Table 7.6. It grows with increasing k_3 and decreasing k_2. When comparing the volumes of the standard Putamen and Cerebellum masks (columns 3 and 5), it changes by a factor of about 2.2. Experiments with other reference volumes (unlisted) showed that the factor is roughly the square root of the quotient of volumes.

Although the table contains unsigned quantities that behave like noise, they instantiate themselves as signed bias for every single patient. This is because there is only one reference curve C_r which pushes k_3 of every voxel in the same direction, upwards or downwards, although with different offsets that depend on k_2 and k_3.

Compared with noise-induced bias of Table 7.6, these figures add an order of magnitude. This makes C_r-induced bias the most significant source of error. As chapters 11 and 14 will show, it plays exactly this role in the real world of MP4A PET.

7.8 Conclusion

Simulations of noisy data are a central feature of this work. They are used to

- compare performance of NLS and COLOGNE
- optimize their weighting strategies
- study noise propagation of NLS with respect to various parameters
- optimize and evaluate mask generating strategies (chapter 14)
- validate voxel based error estimates destined for use with real PET data (chapter 8).

Special care has been taken to model synthetic noise as realistically as possible, adapting its level to every single frame. That noise profile takes account of decay, frame length and the shape of every simulated TAC.

Beyond modelling the *shape* of the noise cloud correctly, we also attempted to assess its overall intensity ("absolute calibration"). It relies on a few parameters built into the simulator: Putamen Decay Fraction, Tanaka formula, Volume Correction Factor. The weakest link in this chain is the Tanaka formula (7.5). It depends on both the scanner and the reconstruction software. We invested a few days to calibrate it for our specific setup, based on a phantom experiment, but results were less than conclusive. Fortunately, absolute calibration does not affect the validation of *k3var* (chapter 8). Of the final results of this work (chapters 15 and 16) only Table 15.5 is affected.

Comparison between *k3dev* (chapter 8) found in Sample 1 images and in simulated data suggests that absolute calibration may be 45 percent low, but such results may also find their explanation in bias of the real data, inhomogeneity of Sample 1 or inhomogeneity of considered regions.

Chapter 8

Voxel based Error Estimates

In the simulator environment, k_3 error assessment is easy: run a large sample of Double Random Simulations and take the empiric variance of computed k_3. If we had a formula estimating that variance based on a single simulated pair of C_t and C_r, it could be used in the real situation as well. Such estimates, *k3var* and *k3dev*, are introduced in this chapter. They are based on linearizing $\mathcal{M}$ around the Model Curve **t** representing the "true" kinetic constants, and modeling the noise cloud in its vicinity (**Figure 8.1**).

For every voxel, the NLS solver provides the distance $\|a - f\| = \|\vec{fa}\|$ of the measured TAC **a** from the set $\mathcal{M}$ of Model Compliant curves. The reason why there is a distance at all, may be threefold:

1. Failure of C_r to properly represent the Blood Input Function. This may be due to both noise and systematic effects.
2. Failure of C_t to be Model Compliant. The reasons are systematic effects like discussed in Chapter 6, or caused by any failure of the biological model or image preprocessing.
3. Noise of C_t.

Presented estimates *k3var* and *k3dev* are based on measuring $\|\vec{fa}\|$ and explaining it entirely with noise of C_t. They assess variance and standard error of k_3 based on how well the measurements C_t fit around the closest Model Compliant curve. Since model-compliance is defined in terms of C_r (section 4.2), they can also be viewed as indicators of compatibiliy between C_r and C_t, which are supposed to depend on the same Blood Input Function.

There seem to be standardized ways of obtaining error estimates for nonlinear regression. But the only reference we were able to find is [34] of 1988, where the concept is generalized to allow errors in the independent variables (which in our context are the framing times). This is not exactly what we need, instead, we wish to include the weighting. We found it easier to develop an independent access (appendix, B). We are still on safe ground, though, since the resulting formula has been validated against the simulator.

8.1 *k3var* and *k3dev*

This section contains a full description of how the estimates are computed. It consists of some initial steps followed by application of formulae (8.2) and (8.4). A detailed derivation is given in appendix B.

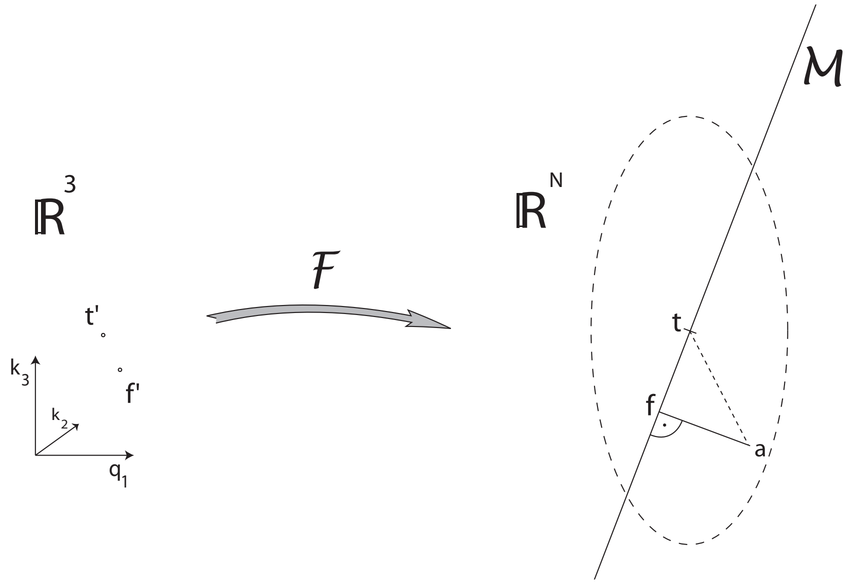


Figure 8.1: *Propagation of noise to error of* k_3

Let **a** denote the measured TAC C_t, f':=(q_1,k_2,k_3) the triple of kinetic constants found by NLS, and **f**=$\mathcal{F}$(f') its Model Curve as in Figure 8.1. The partial derivatives of $\mathcal{F}$ in f' with respect to q_1, k_2 k_3 form a set of vectors in $\mathbb{R}^N$. It will be linear independent if **f** is a regular point of $\mathcal{M}$. Apply the Gram Schmidt orthonormalization procedure to obtain an orthonormal vector system b_1,b_2,b_3:

$$v_1 := \frac{\partial \mathcal{F}}{\partial q_1}(f')$$

$$b_1 := \frac{v_1}{\|v_1\|}$$

$$v_2 := \frac{\partial \mathcal{F}}{\partial k_2}(f') - \left\langle \frac{\partial \mathcal{F}}{\partial k_2}(f'), b_1 \right\rangle b_1$$

$$b_2 := \frac{v_2}{\|v_2\|}$$

$$v_3 := \frac{\partial \mathcal{F}}{\partial k_3}(f') - \left\langle \frac{\partial \mathcal{F}}{\partial k_3}(f'), b_2 \right\rangle b_2 - \left\langle \frac{\partial \mathcal{F}}{\partial k_3}(f'), b_1 \right\rangle b_1 \tag{8.1}$$

$$b_3 := \frac{v_3}{\|v_3\|}$$

where $\langle.|.\rangle$ and $\|.\|$ are defined by equations (4.7) and (4.8) using $w_1,\ldots,w_N$ from equation (2.7) which are the Decay Weights arising from the framing times. Let $s_i := \sqrt{\frac{C_t(i)}{w_i}}$ and $b_{j,i}$ the i'th component of vector b_j (i=1…N; j=1,2,3). Then

$$k3var := \frac{\|\vec{fa}\|^2}{\|v_3\|^2} \cdot \frac{\sum\limits_{i=1}^{N} s_i^2 \cdot b_{3,i}^2 \cdot w_i^2}{\sum\limits_{i=1}^{N} s_i^2 \cdot w_i - \sum\limits_{j=1}^{3} \sum\limits_{i=1}^{N} s_i^2 \cdot b_{j,i}^2 \cdot w_i^2} \tag{8.2}$$

where v_3 is taken from (8.1). This formula assumes an anisotropic N-dimensional normal distribution of the noise cloud and accounts for the inclination at which $\mathcal{M}$ intersects with that cloud. If isotropic noise is assumed instead, it simplifies to

$$k3var_i := \frac{\|\vec{fa}\|^2}{\|v_3\|^2 \cdot (N-3)} \tag{8.3}$$

In either case, *k3dev* is its square root:

$$k3dev := \sqrt{k3var} \tag{8.4}$$

8.2 Validation

For a number of combinations of k_2 and k_3, 10000 Double Random Simulations (section 7.2.1) were performed with standard simulator settings (section 7.2.4). Each pair of C_r and C_t was evaluated with NLS. So we get a sample of 10000 k_3 and 10000 *k3var*. The empiric SD of the former was divided by the square root of the mean value of the latter. The quotients are shown in **Table 8.1**. As *k3var* is an estimator of the variance of k_3, they are expected to be close to 1. Entries larger than 1 indicate underestimation, smaller than 1 overestimation. The top entry of every cell refers to the 674+2 Putamen mask and a realistic tracer dose. We then reduced noise by applying tenfold and hundredfold amounts of tracer in lines 2 and 3. This was expected to clamp down on nonlinearity effects and demonstrate correctness of the formula. While this helped in a few cases, it generally failed in the right part of the table, instead the overestimation got worse. So we returned to 555 MBq and reduced simulated noise only of the reference curves, by raising the reference volume to 16 and 64 ml (in italics). This fixed the problem in all cases.

	k_2					
k_3	0.01	0.02	0.05	0.1	0.2	0.5
0.01		0.940 1.012 1.010	1.013 1.015 1.001	0.996 0.978 0.977	0.914 0.908 0.834 *0.958* *0.982*	0.863 0.851 0.763 *0.937* *0.980*
0.02	0.879 1.005 1.005	0.996 1.006 1.004	0.992 0.994 0.983	0.952 0.952 0.983	0.897 0.892 0.827 *0.950* *0.980*	0.851 0.841 0.760 *0.933* *0.980*
0.05	1.001 1.007 1.006	1.000 1.001 0.999	0.978 0.981 0.973	0.945 0.946 0.923 *0.981* *0.998*	0.904 0.902 0.855 *0.955* *0.981*	0.867 0.860 0.794 *0.944* *0.985*
0.1		1.024 1.002 1.001	0.983 0.985 0.980	0.958 0.960 0.947 *0.985* *0.997*	0.929 0.929 0.900 *0.968* *0.986*	0.900 0.896 0.848 *0.962* *0.993*
0.2			0.990 0.990 0.990	0.981 0.977 0.973	0.962 0.961 0.948	0.940 0.941 0.915
0.3				0.988 0.987 0.984	0.978 0.978 0.970	0.967 0.965 0.950
0.4					0.998 0.988 0.983	0.979 0.980 0.971

Table 8.1: *Quotients of empiric SD of k_3 from 10000 Double Random Simulations and their mean k3dev. In every cell, tracer dosage is set to 555, 5550, 55500, 555 and 555 MBq (from top to bottom), and the reference volume to 5.392, 5.392, 5.392, 16 and 64 ml. NLS evaluation yield of all listed entries was above 90%, keeping selection effects at bay.*

8.3 Averaging of *k3dev*

Averages of *k3dev* over voxel populations or patients of a sample, were computed by the formula

$$\sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} x_i^2} \tag{8.5}$$

in most places of this work, including Table 8.1. This is because, unlike variances, standard deviations are not additive. Exceptions to this rule, for historic reasons, are all data presented in chapters 5 and 10.

8.4 Discussion

By increasing the tracer dosage 100-fold, we reduced the level of noise to one tenth. As the estimates are based on linearization of $\mathcal{M}$, they should profit from noise reduction since it makes $\mathcal{M}$ larger in comparison to the noise cloud, so it comes closer to being linear. However, for high k_2, we found some systematic overestimation which failed to disappear as we reduced the noise. It disappeared only when reducing selectively noise of C_r by increasing the reference volume. In the design of *k3var*, noise of C_r has been ignored. Yet the estimate is sensitive to it, since C_r is part of the definition of $\mathcal{M}$, so $\mathcal{M}$ becomes a “noisy” set which is harder to approximate, increasing $\|a - f\|^2$ in the enumerator of formula (8.2). It even leads to overestimation. So *k3var* works fine in the situation for which it was designed, where noise of C_t is the dominant source of error, and turns out a little too large when noise of C_r is dominant. Overestimation of up to 6 percent was found in situations of practical interest (k_2=0.1 with 555 MBq of tracer), this is good enough for application to real image data.

Chapter 9

The traditional Workflow

Voxel based kinetic analysis (section 1.6) is just one step of a complex workflow leading from raw PET data to parametric k_3 images. This and the following chapter deal with the surrounding procedures of normalization, coregistration, sampling the reference curve, masking and filtering. Much of it is a heritage of G. Zuendorf who implemented the workflow for use in a study of 2003 [14]. His implementation used components of SPM99 [35] which is based on MATLAB6 [28]. We replaced it by C++ code running under VINCI, this will be called the **"conservative MP4A tool"** since it largely follows the Zuendorf policies.

A second implementation is the **"maskless MP4A tool"**, which includes NLS, k_{3r}-correction, *k3var* as additional features of kinetic analysis. Its preprocessing part is based on the Zuendorf procedures, some of which have been replaced or modified by the author. These changes, and their validation, are the subject of Chapter 10.

9.1 Normalization

Mean values of k_3 for anatomical regions of the brain (such as is Table 13.1) are the final goal of analysis, but first the regions need to be defined. As of today, nobody has time for manually outlining them on individual images, based on anatomical structures visible on MRs of the same patient. An alternative is using a set of fixed masks. Such a set is called an **atlas**, and provided in the form of an image of numeric labels, one for each region of the brain. To make the use of an atlas possible, the parametric images need to be in the same coordinate space as the atlas. Transformation into this space is called "normalization". It intends to ensure that, for instance, the "Hippocampus" mask really covers the Hippocampus of the patient. This is not quite easy since patient anatomies are slightly different, and cannot always be fully achieved by automatic procedures [6].

9.2 From Acquisition to Evaluation Space

In both MP4A tools, image preprocessing and kinetic analysis take place in the coordinate space of the atlas, which will therefore be called **Evaluation Space**. Raw PET images are transformed from their own format (**Acquisition Space**) to Evaluation Space. These steps include correction for patient movement: as people find it hard to stay in exactly the same place for 60 minutes, there will be spatial differences between the frames that must be undone in order to get meaningful TACs.

Figure 9.1 presents an overview of the full workflow. Images in Evaluation Space are presented in squarish shapes, images in Acquisition Space in flat shapes. The acquisition format depends on the scanner and its software. On most scanners, resolution in z direction (which is the patient's longitudinal axis) is lower than in x and y direction. So in z direction the number of layers is reduced and the voxel size increased.

Transformation from Acquistion to Evaluation Space requires a **Normalization Template** onto which the acquisition frames are normalized. It is an image in Evaluation Space, congruent with the atlas. Its modality[1] should match the PET image we wish to normalize. In our case, that modality is called **"flow"** or **"perfusion"**, it is the volume of blood delivered to 1 ml of tissue per second. Every tracer with a high extraction rate[2] produces a perfusion image in the first few minutes. This is because its initial uptake is proportional to delivery by the blood. This applies also to MP4A.

9.2.1 Early Frames

The set of acquisition frames is divided in two packages, called "**Early Frames**" (together covering the first 10 minutes of scantime) and "**Late Frames**". Both are transformed from Acquisition to Evaluation Space in a single step. The transformation must be mediated by a so-called **Flow Image** for two reasons:

1. Image Modality. Normalization to the template is an inter-subject step, it therefore requires the same image modality (section 2.10.1). We are therefore restricted to the time window during which MP4A images are still perfusion images. This is no longer the case after the first 10 minutes: for instance, Putamen and Cerebellum are not very eye-catching on the Early Frames, but end up being the brightest regions later.

2. Signal strength. Automatic normalization also requires a sufficient

[1] The modality of a PET image is usually characterized by the tracer, although this example is slightly different

[2] the percentage of uptake in the tissue during the first passage through the capillary system

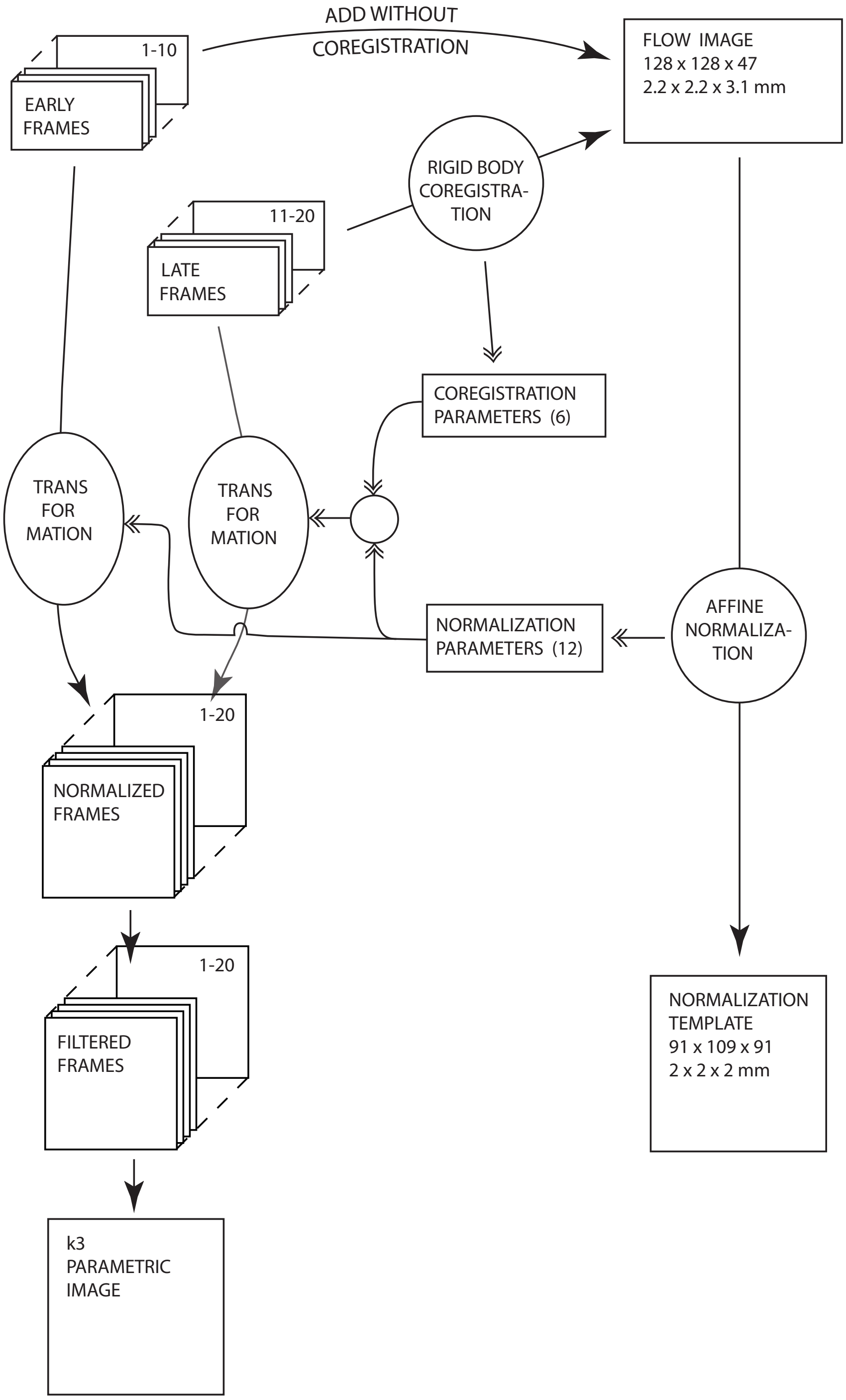


Figure 9.1: *Image Preprocessing Workflow*

amount of viable image detail, which implies a decent signal-to-noise ratio. This is not provided by an Early Frame in isolation, since it covers a short time interval and therefore has poor signal strength. So we need to add up a few of them to get enough statistics for the automatic procedure to work.

The Early Frames are added without coregistration, so we rely on the patient not having moved during the first 10 minutes. Then we run Affine Normalization as implemented in VINCI with 12 degrees of freedom. The resulting image is discarded, but the 12 parameters are collected. They characterize the affine function f that aligns the Flow Image with the template, and we use this function to transform each of the Early Frames in the same way.

9.2.2 Correction for Movement

The Late Frames that cover the remaining 50 minutes, are corrected for patient movement. They have sufficient signal strength to be matched with the Flow Image. Since this is intra-subject coregistration, we can use the more robust "Rigid Body Coregistration" method implemented in VINCI, with 6 degrees of freedom (section 2.10).

The coregistered frames could, in principle, be mapped to Evaluation Space by the same function f that was used for the Early Frames. However, this would require two transformations for every Late Frame: coregistration followed by application of f. We can save some image quality by concatenating the two transformation steps into one. From the coregistration parameters of every Late Frame, and the normalization parameters of the Flow Image, we compute the transformation matrix of the combined mapping. So we need to resample the images only once.

9.3 Image Preprocessing

In order to reduce noise before sampling voxel based TACs, the normalized frames are smoothed by Gauss filtering. While this improves the quality of the TACs, it also sacrifices spatial resolution - a tradeoff to be discussed in detail in the next chapter. In particular, if a region has moderate signal strength and lies next to the hot Basal Ganglia, its signal is corrupted by influx from the neighborhood during the Gauss Filter step. For this reason, we apply protective masking.

9.3.1 The Zuendorf Mask

was generated by G. Zuendorf for use with his MATLAB6 implementation. It selects the set of voxels that had shown the most robust kinetic results on a sample of patients. Its size is 115471 voxels, covering 45.9% of the normalized brain, mainly cortex regions. It fully removes the Basal Ganglia and the Cerebellum.

9.3.2 Masking and Border Correction

Thus, the set of “nice” voxels is protected against influx from hot neighborhoods prior to Gauss filtering. The price is an artificial internal border. Instead of influx from hot neighborhoods, we now have influx from the surrounding zero level, which results in darkening of the filtered images close to the mask borders. For a given voxel, this will reduce intensity by about the same factor on all frames. By Theorem 4.5, such scaling is neutral with respect to k_2 and k_3, while fully affecting q_1.

We therefore correct for the darkening by dividing every voxel through a reference intensity. That intensity is obtained by subjecting the Zuendorf Mask - consisting of all ones, surrounded by zeroes - itself to Gauss filtering, and picking the value of the corresponding voxel.

9.4 Obtaining the Reference Curve

While the voxel based TACs C_t are sampled from the masked filtered frames, C_r (see section 3.2.2) is obtained from the unfiltered normalized frames, using a “reference mask”. That mask (or rather, two of them, for Putamen and Cerebellum) were compiled by G.Zuendorf, presumably by some kind of intensity thresholding on the average of the sum images of a sample of patients. They cover 5.392 ml (Putamen) or 117 ml (Cerebellum). As the intention was to cover just the hottest voxels, the Putamen mask is less than half as big as the corresponding brain region.

Thus, the same mask is used for all patients, trusting the normalization and coregistration steps to ensure that it fits on its target.

9.5 Critique

While it is pretty safe to do this for Cerebellum, on the Putamen there is considerable risk that the region is missed. In addition to being small, it is also divided in two components (left and right Putamen). Differences in anatomy may already lead to a mismatch, not to mention failed coregistration/normalization.

Similar problems were encountered with the Zuendorf Mask. It is designed as a compromise between masking out the problematic regions, and retaining as much of the brain as possible. But since one mask is used for all patients, it may fall short on both purposes. While only 45.9% of the brain are covered, one could often see a hot region shine through at the border, resulting in hot spots on the k_3 images. We therefore decided to switch to an implementation with adaptive masking.

Chapter 10

The maskless Workflow

This chapter deals with the “maskless MP4A tool”, which is the second implementation in VINCI. Much of the workflow has been restructured. The objective was to

- use no fixed masks for cortex and reference regions, in order to avoid the problems laid out in section 9.5
- decrease the amount of brain that was previously masked out
- optimize image preprocessing with respect to precision of k_3.

10.1 Image Resources

The resources used with the maskless tool are based on an H_2O PET template shipped with SPM99 [35]. “Evaluation Space” (section 9.2) is defined by this template ($91 \times 109 \times 91$ voxels sized $2 \times 2 \times 2$ millimeters) . The atlas has been downloaded from LONI [26], where it is called “ICBM Template”. Its format is $181 \times 217 \times 181$ voxels sized $1 \times 1 \times 1$ mm. When PET.img was compared to this atlas in VINCI, the two images were perfectly congruent. We therefore resliced the atlas to Evaluation Format (section 9.2) using VINCI’s nearest-neighbor interpolation.

To get rid of all non-brain voxels, we also needed a **full brain mask**. This was generated from the file EPI.img of the SPM99 package, which is in Evaluation Format. It represents a MR modality having the helpful property that brain is distinctly brighter than the rest of the scull, with the exception of the eye balls. Thus by intensity thresholding one can obtain a brain-plus-eyes mask. The threshold was chosen so as to let a certain fraction of the voxels become part of the mask. After some experimenting, that fraction was set to 28.1%. In the resulting mask, the eye balls were isolated connected components, which made it easy to remove them.

One would now expect the atlas to be a subset of the full brain mask. This was almost the case, except for 1412 atlas voxels extending beyond the mask. These were subsequently added. Now the mask had 251461 voxels, representing a volume of 2011.688 ml.

10.2 Erosion, Dilation, Thresholding

These image processing techniques are relevant in the context of this chapter. **Thresholding** is used for generating voxel sets, to be stored as binary masks. They consist of all voxels whose intensity in a given image is above or below a scalar value called **threshold**, or between two thresholds. As reference we use an image of **Overall Intensities**, by which we understand the Decay Weighted sum of all normalized frames (section 2.7.2). Such an image is shown in the top row of Figure 10.1.

Dilation and **erosion** are used in their simplest form for growing or shrinking of voxel sets. Every voxel has between 3 and 6 **direct neighbors** in the volume (from which it differs by 1 in one coordinate). A single dilation cycle grows a given voxel set by adding the direct neighbors of all voxels. Conversely, a single erosion cycle removes the direct neighbors of all voxels of the complement set. Applying erosion and dilation cycles in succession does usually *not* reproduce the original set, it is easy to see that, for instance, isolated voxels are thus removed. Applying them in reverse order results in filling up isolated “holes”.

10.3 Reference Mask

For reasons given in section 9.5, we no longer use the fixed masks of section 9.4. Instead, we start with the Putamen and Cerebellum masks contained in the aforementioned atlas. They consist of 1557 (23229) voxels or a volume of 12.456 (185.832) ml. However, they are only used as mask precursors, from which the mask proper is computed for every patient individually. Such computation involves parameters named **‘padding’**, **FWHM2** and **‘regionsize’**. Setting ‘padding’ to an integer n causes successive application of n dilation cycles if n is positive, or $|n|$ erosion cycles if n is negative, to the precursor mask.

Thereafter, a number of ‘regionsize’ voxels are selected from the resulting set to make the final mask. They are the voxels of maximum Overall Intensity. In order to keep the resulting set compact, the intensity image is Gauss-filtered prior to selection. The filter width, FWHM2, was set to 4 millimeters.

10.3.1 Standard Masks

Two standard policies of mask generation were used for this work. The **Putamen 674+2 Mask** applies ‘padding’=2 and ‘regionsize’=674. The padding makes it possible to obtain a perfect mask even if the precursor misses the Putamen by up to 4 millimeters. The number 674 stems from the Zuendorf Mask of section 9.4. Its volume of 5.392 ml is the expected size of the "hot" portion of Putamen. Whenever it was optically verified, the Putamen 674+2 masks were found to be compact and fit nicely on the hottest Putamen voxels of the sum image.

The **Cerebellum 3500-2 mask** applies 'padding'=-2 and 'regionsize' =3500. The negative padding serves to exclude all voxels lying too close to some other atlas region, in particular Cerebrospinal Fluid. Restricting the mask size to a fraction of the Cerebellum is another attempt to evade Partial Volume Effects and improve quality: it seems safer to select the hottest voxels, than expecting the literature value of k_3=0.65 (section 13.1) to apply to the whole Cerebellum. These masks were reasonably compact owing to the Gauss Filter step, but they extended over much of the cerebellar cortex forming irregular patterns that differed from individual to individual.

10.4 Selecting Voxels

In the previous chapter, the Zuendorf Mask was the answer to all questions on voxel selection. It covers the 45.9% of the brain that can be expected to evaluate to reasonable quality with the COLOGNE method. k_3 computation was attempted only on that set, and the number of failures never exceeded a few hundred voxels. As a consequence, all k_3 images had the shape of the mask, making it easy to compare them across a sample of human subjects. There are three types of selection to be considered:

- **preselection**: removing voxels before Gauss filtering
- **failselection**: loss of voxels due to failure of kinetic analysis
- **postselection**: removal of untrustworthy voxels after kinetic analysis.

Preselection by the Zuendorf Mask is the only selection applied in the traditional workflow. Since it occurs before the Gauss filtering step, it affects the shape of the TACs and thereby affects the k_3 results. Failselection will occur inevitably if preselection is omitted, typically removing some 10% of all voxels. It creates a different image shape in every patient. Postselection is an interesting idea as it could benefit from the information gathered in the meantime, in particular since quality markers like *k3dev* (Chapter 8) and *uff* (section 4.7.4) are now available. It is, however, problematic because it introduces a new difficulty:

10.4.1 Selection Bias

When the pattern of removed voxels is different for every patient, *every* type of selection introduces its own bias. The reason is that almost no type of selection (except random selection) will be fully orthogonal to k_3. For instance, failselection preferably eliminates voxels with high k_3, since they are harder to evaluate. The same is true for postselection based on *k3dev* thresholding: high k_3 leads to high *k3dev*, so voxels with high k_3 are more likely to be sorted out. This leads to a shift of the average k_3, rendering regional results untrustworthy. The only remedy is to keep selection as low as possible, and exclude regions from evaluation that have lost a certain percentage of their voxels.

In the traditional workflow, it had been partly justified to ignore the problem of Selection Bias: of course the Zuendorf Mask induces such bias to all regions which are cut by its borders, however as the part under the mask is identical for all patients, inter-subject comparability is preserved.

10.4.2 Adopted Selection Policy

To ensure reliablity of regional results, we will therefore

1. Avoid pre- and postselection alltogether

2. Keep the yield after failselection as high as it is possible without compromising the quality of k_3 results

10.4.3 Intensity Zoning

The traditional workflow includes Gauss filtering for reduction of noise, and protective masking during the filter step in order to avoid contaminating the cortex with signal from neighboring hot regions. Both measures are preserved in the maskless MP4A tool.

The Zuendorf Mask is replaced by a sequence of masks. They are computed from the normalized frames, and we introduce a number of parameters to control their generation and application.

The masks are created by multiple thresholding for Gauss-filtered Overall Intensity (section 10.2). Thus we start with a Decay Weighted sum image such as shown in Figure 10.1, top row. Gauss filtering is applied before thresholding in order to render the masks as compact as possible. Its width **FWHM3** is one of the tuning parameters, it was set to 8 millimeters.

The thresholds form a geometric sequence, whose quotient **‘qzoning’** was set to 0.85. The first threshold is ‘qzoning’ times the intensity of the brightest voxel; it creates the largest mask, containing all voxels of lower intensity. The thresholds and their masks are then getting smaller, and we stop just before a mask covers less than two percent of the brain.

The smallest mask, containing the faintest voxels, is called mask 1. Masks are numbered consecutively, the largest index is between 10 and 15, depending on the input. The difference sets of the masks are called **zones**. Zone 1 equals mask 1, zone 2 consists of the voxels of mask 2 minus those of mask 1, and so on up to the highest zone which is the full brain minus the highest mask. The zones thus become disjoint sets whose index reflects increasing intensity, defining a subdivision of the brain as illustrated in Figure 10.1, bottom row.

Every voxel **v** of the brain is located in exactly one zone. During the filter step, it is isolated in a **filter mask** containing its zone and all zones of lower index, corresponding to lower intensities. Using this mask, Gauss filtering and border correction are applied in the same way as in the traditional workflow.

We thus have the same protection effect as by the Zuendorf Mask, in that voxels of higher zones are prevented from influencing **v**. But unlike in the conservative MP4A tool, they are not removed. They undergo filtering provided by their own filter masks, and the results of all voxels are collected in one image. Or rather, in a sequence of images, called the **masked filtered frames**, since every normalized frame undergoes the same procedure.

'qzoning' is set to 0.85, so the zones are not very wide. Therefore, each voxel ends up close to the inner border of its filter mask. That effectively removes about half its neighborhood. Some voxels end up in a small island or peninsula, where their isolation leads to loss of signal from the neighborhood, resulting in a noisy TAC and poor evaluation. So the amount of masking should be carefully controlled.

A new parameter **'margin'** is introduced for this purpose. With 'margin' set to 0, the masking policy is as described above. 'margin'=1 adds one zone to every filter mask. For instance, for voxels of zone 3, the filter mask will be the union of zones 1 to 4.

Higher values of 'margin' add several zones. For most voxels this means, they are no longer close to the border. This fixes the aforementioned problems. Only voxels near a steep gradient end up close to the border, in accord with the idea of shielding them from hot neighborhoods.

Note that the zoning policy provides for the same protection that was previously achieved via preselection. In addition, it can be tuned by 'qzoning' and 'margin'. Setting 'margin' to 0 results in maximum masking, while setting it to the number of zones results in no masking at all.

'qzoning' should be close to 1 in order to provide a sufficiently fine subdivision. The parameter was introduced to keep something in hand to prevent the formation of artefact "stripes", reminding of a lawn mower. If that had occurred, we could have raised 'qzoning' in order to get finer granularity. However, such artefacts were rarely seen with 'qzoning'=0.85.

The remainder of this chapter is devoted to studying the effect of 'margin' and FWHM on the quality of k_3 results.

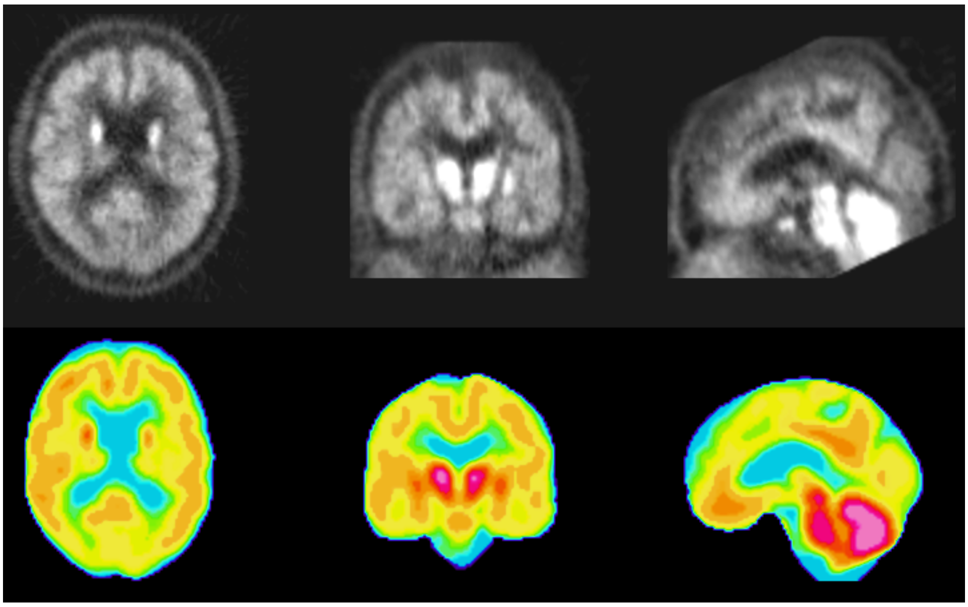


Figure 10.1: *Decay Weighted Sum (above) and zoning image (below) of subject M01007. From the sum image, the brain was isolated and subdivided in 13 zones, whose properties are explored in chapters 10 and 12. Zone 1 is in blue, zone 13 in bright magenta.*

10.5 Zonal Results

will be used in the following sections to study the effects of 'margin' and FWHM. They are the mean values of all voxels of a zone. Depending on the image modality, we can have zonal k_3 results, *k3dev* results, etc.. Voxels that fail to evaluate are excluded from statistics. The percentage of good voxels (to be called **yield** or **yield after failselection**) is recorded, in order to keep an eye on the risk of Selection Bias (section 10.4.1).

10.5.1 Privileged Area

To render results more meaningful, we exclude noisy outliers from statistics. This selection is made using a fixed mask called **Privileged Area**. It is the union of all atlas regions promising reasonable results (appendix A.4), in terms of high yield and low *k3dev*. It contains 68,37% of the brain, mostly cortex areas. Roughly, it can be viewed as a superset of the Zuendorf Mask. Note that it is only used for statistics, not as part of the workflow.

10.6 Optimizing the Parameters

In the absence of preselection, the amount of masking and filtering is governed by ‘margin’, ‘qzoning’ and FWHM. For ‘qzoning’, the intensity quotient between the zones, a value of 0.85 is close enough to 1 to provide a sufficiently fine gradation. It typically resulted in 12 to 14 intensity zones. ‘margin’ controls the influx from intense toward less intense zones. In the images of subject M01007, there were 13 zones. Setting ‘margin’ to 12 disables all masking there, such that Gauss filtering is performed on the full brain.

All computations in this chapter were carried out on the PET data of subject M01007. **Table 10.1** shows some global effects of ‘margin’. In column 2, we see the percentage of brain voxels for which k_3 could be computed. The number of voxels that were successful for *all* listed values of ‘margin’ was 217908, which is 86.7% of the brain. Columns 3 and 4 show mean values of two quality markers, taken on this “total intersection set”. In columns 5 to 8, the same statistics is shown for the Privileged Area: the percentage of voxels where k_3 could be computed, followed by mean values on the corresponding intersection set. This **“Privileged Intersection Set”** was used for all validations of this chapter. It consists of the voxels of the Privileged Area that evaluated successfully for ‘margin’=0,1,2,3,4 and 12. These were 166143 voxels, 96.64% of the Privileged Area.

		on total		on Privileged Area			
		intersection set			on Privil. Intersection Set		
‘margin’	yield	*k3dev*	*uff*	yield	*k3dev*	*uff*	k_3
0	89.1%	.0290	.5971	97.04 %	.0153	.6044	.0785
1	90.6%	.0262	.6036	98.16 %	.0125	.6100	.0790
2	91.2%	.0255	.6012	98.64 %	.0118	.6087	.0793
3	91.5%	.0252	.5984	98.85 %	.0116	.6074	.0794
4	91.5%	.0252	.5970	98.96 %	.0115	.6069	.0794
12	91.7%	.0251	.5964	99.02 %	.0115	.6067	.0794

Table 10.1: *Global yield and quality markers*

10.6.1 ‘margin’

will be looked at first, while keeping FWHM at 8mm. For *k3dev* and ’yield’ in Table 10.1, we see steady improvement as the amount of masking is reduced, most pronounced in the beginning and quickly reaching saturation as ‘margin’ approaches 12. In the Privileged Area, we see better scores on all quality markers (columns 5, 6, 7 compared to 2, 3, 4). This comes as no surprise since the area has been selected for containing “nice” voxels. *k3dev* can be viewed as a measure of how much NLS approves of its input curves. Apparently, these are getting better as we reduce the amount of masking and allow further influx from high intensity zones to lower ones. However, while NLS gets increasingly happy, we do not know if the computed k_3 are still representative of the underlying brain voxel. To find out, we use the "unfiltered feedback" marker of section 4.7.4. Its columns 4 and 7 are the only ones showing deterioration, after going through an optimum at 'margin'=1.

k_3 itself (column 8) is almost constant. It would therefore be nice to find out how much difference at all there exists between k_3-images obtained with different values of 'margin'. To keep outliers out of play, we restrict this comparison to the Privileged Intersection Set: Image-image correlation and Euclidean distance (section 2.8) are listed in **Table 10.2**. Obviously, the image that differs most from the rest is computed with 'margin'=0. Between 'margin' 2,3,4 and 12, there is almost no difference. So we can limit investigations to 'margin'=0, 1, and 12.

'margin'	1	2	3	4	12
0	0.8729	0.8275	0.8229	0.8210	0.8205
	0.0204	0.0240	0.0243	0.0244	0.0245
1	·	0.9731	0.9709	0.9692	0.9687
	·	0.0095	0.0098	0.0101	0.0102
2	·	·	0.9977	0.9969	0.9964
	·	·	0.0010	0.0012	0.0013
3	·	·	·	0.9997	0.9994
	·	·	·	0.0009	0.0014
4	·	·	·	·	0.9999
	·	·	·	·	0.0007

Table 10.2: *Global Similarities of k_3-images:* 'corr' *(upper number) and* 'dist' *(lower number)*

Column 2 of **Table 10.3** gives the distribution of the full brain to the zones, it adds up to 100%. Column 3 shows the same for the Privileged Area. The zones of maximum intensity, 11,12 and 13, cover only 0.35% of it. So they are of little practical interest.

The right part of the table shows zonal mean values of *k3dev* on the Privileged Intersection Set, for images computed with 'margin' = 0, 1 and 12. By far the lowest readings are seen in zones 7 and 8, followed by zone 6. Together they cover 61.3% of the brain, and 76.6% of the Privileged Area.

zone	*%of* *brain*	*%of* *Privil.Area*	*k3dev* 'margin'=0	 'margin'=1	 'margin'=12
1	3.02	1.15	0.1023	0.0619	0.0351
2	2.54	1.33	0.0628	0.0368	0.0209
3	4.03	2.62	0.0383	0.0247	0.0158
4	6.15	4.85	0.0256	0.0184	0.0132
5	9.23	8.92	0.0201	0.0143	0.0116
6	15.95	18.29	0.0153	0.0112	0.0104
7	25.82	33.19	0.0117	0.0099	0.0098
8	19.51	25.07	0.0100	0.0097	0.0098
9	4.41	3.37	0.0211	0.0235	0.0254
10	2.96	0.86	0.0446	0.0474	0.0494
11	2.78	0.29	0.0601	0.0690	0.0693
12	2.67	0.06	0.1280	0.0894	0.0894
13	0.94	0.00	-	-	-

Table 10.3: *Zonal properties and k3dev*

Apparently, these **Medium Zones** are of best signal quality. It deteriorates sharply toward the **High Intensity Zones** 9 to 13, which account for 4.58% of the Privileged Area. The **Low Intensity Zones** cover 18.87%. There too, we see a gradual decline in quality from zone 5 to 1. As the comparison between the columns shows, there is much to be gained by allowing signal influx from higher zones. So, the poor quality in zone 1 is most likely a result of low signal strength, which leads to noisy TACs. The situation is reversed in zones 9 to 11, where image quality apparently benefits from the masking.

Next we localize the k_3 differences between 'margin'=0 and 12 by restricting image comparison to the zones: **Table 10.4** shows the similarity measures of section 2.8. In the left part, we find analogies to the previous table: the superior quality of the Medium Zones, the sharp deterioration from zone 8 to 9, and steady deterioration from zone 5 down to 1.

Yet there is also a new indentation in zone 9, which shows greater differences between the images than zones 8 and 10. There is a conspiciously high reading (bold print) in the 'shift' column of zones 9 and 10. As these zones are bordering the reference regions, they are the ones that motivated masking in the first place, and we see the direct effect of it - it prevents a systematic rise of k_3.

In the lowest zones, there is even more shift, caused by a drop of k_3 (unlisted) from 0.1135 to 0.0869 in zone 1 and 0.0916 to 0.0723 in zone 2. It can be blamed on low signal strength and noisy TACs, resulting in unstable and biased evaluation. This interpretation is corroborated by many other data. One example is the low yield after failselection in these zones (unlisted). For ‘margin’=0, it is 30.4% in zone 1 and 60.9% in zone 2. With ‘margin’=1, it goes up to 44.9% and 80.9%, to reach 75.5% and 95.7% for ‘margin’=12. This is indicative of poor TACs for ‘margin’=0, and getting better as influx from neighboring zones is allowed. It is easy to picture this effect when bearing in mind that zones 1 and 2 form a thin layer at the periphery of the brain, as visible in Figure 10.1. Similar evidence from the *k3dev* marker has already been shown in Table 10.3.

It remains to decide which ‘margin’ leads to better k_3 results. So we look at the *uff* marker (“**Unf**iltered **F**eedback”, see section 4.7.4). Its interpretation is delicate. Since it also responds to unspecific noise, it increases with the zonal index, as discussed in section 4.7.4. So we can use it only for comparison within a given zone.

In **Table 10.5**, we find ‘margin’=1 beat ‘margin’=0 in every zone. Interestingly, it also dominates over ‘margin’=2 and 12 in zones 1 to 9. This has already been seen in the global statistics of Table 10.1. In the highest zones, the differences tend to level out. This is because no filtering at all takes place in higher zones for higher ‘margin’.

10.6.2 FWHM

FWHM is the size of the Gauss Filter applied during image preprocessing (section 9.3). Like ‘margin’, it controls how much signal a voxel receives from its neighborhood. The difference is that Gauss filtering is isotropic in space, while ‘margin’ deals with signal flow across (and therefore orthogonal to) zonal borders.

	comparing ‘margin’ 0/12			comparing ‘margin’ 1/12		
zone	‘corr’	‘dist’	‘shift’(0-12)	‘corr’	‘dist’	‘shift’(1-12)
1	0.6566	0.0853	0.0265	0.8882	0.0459	0.0094
2	0.5328	0.0721	0.0193	0.8259	0.0321	0.0039
3	0.6059	0.0497	0.0070	0.8602	0.0233	0.0018
4	0.7658	0.0314	0.0015	0.9148	0.0143	0.0000
5	0.7859	0.0234	-0.0006	0.9526	0.0090	-0.0011
6	0.8494	0.0173	-0.0016	0.9767	0.0055	-0.0005
7	0.9008	0.0101	-0.0008	0.9898	0.0030	-0.0002
8	0.9034	0.0127	-0.0010	0.9800	0.0061	-0.0004
9	0.7363	0.0790	**-0.0138**	0.9557	0.0340	**-0.0043**
10	0.9543	0.0660	**-0.0105**	0.9974	0.0183	**-0.0047**
11	0.9318	0.0656	-0.0008	0.9976	0.0128	-0.0001
12	0.9739	0.0604	-0.0109	1.0000	0.0000	0.0000

Table 10.4: *Zonal similarity measures*

zone	‘margin’=0	‘margin’=1	‘margin’=2	‘margin’=12
1	0.2008	0.2027	0.1972	0.1758
2	0.2937	0.2968	0.2888	0.2539
3	0.3680	0.3730	0.3631	0.3337
4	0.4430	0.4528	0.4456	0.4321
5	0.5083	0.5198	0.5150	0.5120
6	0.5726	0.5818	0.5809	0.5806
7	0.6300	0.6346	0.6346	0.6345
8	0.6773	0.6785	0.6785	0.6783
9	0.6811	0.6856	0.6849	0.6841
10	0.6967	0.7146	0.7150	0.7144
11	0.7241	0.7481	0.7486	0.7486
12	0.7567	0.7785	0.7785	0.7785

Table 10.5: *Zonal uff*

We examine the effect of FWHM on *k3dev* for ‘margin’=0, 1, 2 and 12 on a subset of the zones. **Table 10.6** is subdivided in blocks, and in each block FWHM is set to 6, 8 and 10 millimeters. As the Gauss Filter step is meant to reduce noise, we expect *k3dev* to improve in every block from top to bottom. This is indeed the case in the Medium and High Intensity Zones. In the low zones, there are three exceptions for ‘margin’=0 and 1, where the reading is worse for FWHM=8mm than for 6mm. Allowing radial influx (by increasing ‘margin’) apparently helps the image quality there, and also “normalizes” the blocks’ behavior with respect to FWHM.

Table 10.7 shows the same investigation with respect to the *uff* marker. While it doesn’t allow comparison between the zones, it is valid for comparisons within the same zone. The middle entries of every block have already been listed in Table 10.5. Unlike with *k3dev*, we see no increase in quality from FWHM=6 to 10mm. To the contrary, in many blocks (bold print) the 6mm reading is best. This was not the case in the unlisted zones 9 to 12. In all zones, the 10mm result was inferior to 8mm. The interpretation is easy. While more Gauss filtering leads to smoother TACs, with better model compliance and therefore easier to evaluate, at the same time k_3 becomes less representative of the underlying area. There is a tradeoff between spatial resolution and reduction of noise. FWHM should be chosen as low as the signal strength permits.

In the Medium Zones where we have excellent signal, we can afford the highest spatial resolution and go as low as 6mm. In the Low Intensity Zones, 6mm is affordable only by raising ‘margin’, in order to bolster signal strength by allowing influx from higher zones.

zone	FWHM[mm]	‘margin’=0	‘margin’=1	‘margin’=2	‘margin’=12
	6	0.0759	0.0603	0.0532	0.0490
1	8	0.1023	0.0619	0.0465	0.0351
	10	0.0798	0.0518	0.0381	0.0227
	6	0.0525	0.0440	0.0378	0.0328
2	8	0.0628	0.0368	0.0275	0.0209
	10	0.0476	0.0297	0.0212	0.0140
	6	0.0171	0.0149	0.0149	0.0149
7	8	0.0117	0.0099	0.0098	0.0098
	10	0.0086	0.0072	0.0072	0.0072
	6	0.0148	0.0144	0.0145	0.0145
8	8	0.0100	0.0097	0.0098	0.0098
	10	0.0073	0.0071	0.0072	0.0072
	6	0.0296	0.0324	0.0327	0.0328
9	8	0.0211	0.0235	0.0253	0.0254
	10	0.0144	0.0158	0.0171	0.0173

Table 10.6: *k3dev depending on ‘margin’ and FWHM*

10.7 Conclusion

A new strategy for image preprocessing was developed, guided by ideas of the traditional workflow: smoothing combined with protective masking. It involves no voxel preselection and requires no fixed masks.

The procedure is controlled by four tuning parameters, FWHM3, ‘qzoning’, FWHM and ‘margin’. Appropriate values for FWHM3=8mm and ‘qzoning’=0.85 were easily found. For FWHM there is a tradeoff between noise reduction and spatial resolution. A similar tradeoff exists for ‘margin’ which controls the amount of masking. Severe masking (as with ‘margin’=0) was found to cause problems in areas of low signal strength, such as the periphery of the brain, while its effects were beneficial to voxels near hot regions. ‘margin’=1 is a sensible compromise. It was found to maximize the *uff* marker, an indicator of the match between the computed kinetic constants of a voxel and its TAC *before* image preprocessing.

With this marker, it was also possible to establish FWHM=8mm as a feasible Gauss Filter size. In areas of good signal strength it is possible to

zone	FWHM[mm]	'margin'=0	'margin'=1	'margin'=2	'margin'=12
	6	0.1571	0.1762	0.1841	**0.1816**
1	8	0.2008	0.2027	0.1972	0.1758
	10	0.1789	0.1958	0.1912	0.1536
	6	0.2443	0.2826	0.2868	**0.2743**
2	8	0.2937	0.2968	0.2888	0.2539
	10	0.2798	0.2913	0.2830	0.2291
	6	0.3372	0.3658	**0.3641**	**0.3545**
3	8	0.3680	0.3730	0.3631	0.3337
	10	0.3592	0.3674	0.3570	0.3076
	6	0.4199	0.4524	**0.4518**	**0.4481**
4	8	0.4430	0.4528	0.4456	0.4321
	10	0.4348	0.4478	0.4391	0.4141
	6	0.5009	**0.5238**	**0.5236**	**0.5230**
5	8	0.5083	0.5198	0.5150	0.5120
	10	0.5003	0.5146	0.5070	0.5003
	6	**0.5753**	**0.5885**	**0.5885**	**0.5884**
6	8	0.5726	0.5818	0.5809	0.5806
	10	0.5644	0.5764	0.5744	0.5737
	6	**0.6359**	**0.6397**	**0.6397**	**0.6397**
7	8	0.6300	0.6346	0.6346	0.6345
	10	0.6233	0.6299	0.6298	0.6296
	6	**0.6817**	**0.6831**	**0.6831**	**0.6831**
8	8	0.6773	0.6785	0.6785	0.6783
	10	0.6713	0.6731	0.6729	0.6726

Table 10.7: *'uff' depending on 'margin' and FWHM. Bold print is used to highlight blocks where the 6mm reading is best*

go as low as 6mm.

The parameters are good for Sample 1 and 2 images that were acquired in Cologne. They need reevaluation when the scanner or the protocols are changed. For instance, PET scans of Samples 3 and 4 (from Milan), when filtered with FWHM=5mm, led to parametric k_3 images that looked like

Sample 1 images after filtering with FWHM=8mm.

Note that ‘margin’ and ‘qzoning’ are interconnected. Moving ‘qzoning’ closer to 1 may require to raise ‘margin’, too.

Chapter 11

Reference Curves

Reference curves (section 3.2.2) provide the information on a patient's Blood Input Function. They are sampled from the unfiltered normalized frames using a reference mask. The process of mask generation is explained in section 10.3, it produces a Putamen or Cerebellum mask of prescribed size while adapting to anatomical differences of the subject. In analogy to the Zuendorf approach, we set the mask size to 5.392 ml during the early phase of this work. A Cerebellum mask of 28 ml was used for comparison. Both sizes are larger than the equivalent volume (section 7.1.5) of 1.75 ml that determines the noisiness of voxel based TACs after 8mm Gauss filtering. We therefore anticipated no problems arising from noisy reference curves. This changed dramatically when the first regional results of Sample 1 patients were analyzed.

11.1 Polymorphism of C_r

11.1.1 Comparisons based on Regional Results

Regional k_3 results (i.e. mean values of all voxels of a region) of Sample 1 patients are listed in **Table 11.1**. They refer to Superior Frontal Gyrus and Hippocampus, and have been sampled from k_3 images obtained with reference curves from aforementioned Putamen and Cerebellum masks. Considerable differences between the columns are seen at first glance. When quantified using the 'dist' measure of section 2.8, they amount to 0.0059 for the gyrus and 0.0157 for Hippocampus.

Column 4 is a pattern of P's and C's indicating which reference curve leads to higher k_3. It applies to both parts of the table. So what bias the reference curves inflict on k_3, is reflected in the same way in both regions.

	Sup. frontal Gyrus			Hippocampus	
patient	Putamen	Cerebellum		Putamen	Cerebellum
M01001	0.1030	0.0922	P	0.1287	0.1111
M01002	0.0976	0.0938	P	0.1062	0.0990
M01003	0.0867	0.0927	C	0.1173	0.1326
M01004	0.0885	0.0849	P	0.1349	0.1321
M01005	0.0818	0.0773	P	0.1471	0.1327
M01007	0.0808	0.0749	P	0.1380	0.1228
M01008	0.0826	0.0866	C	0.1221	0.1324
M01009	0.0682	0.0757	C	0.0883	0.1014
M01010	0.0842	0.0881	C	0.1359	0.1651
M01012	0.0814	0.0783	P	0.1307	0.1218
M01013	0.0936	0.0893	P	0.1177	0.1049
M01014	0.0951	0.0873	P	0.1274	0.1044

Table 11.1: *Comparing Putamen and Cerebellum reference curves based on mean values of k_3, averaged across two regions of the brain*

11.1.2 Comparisons based on Simulations

But the consistency ends when comparing C_r of different patients: for instance, M01013 has higher readings than M01008 in Superior Frontal Cortex, while in Hippocampus the order is reversed. This is not surprising, since we expect to find different regional k_3 patterns in different subjects.

To render the reference curves comparable, we now exclude any influence of the target regions and use simulated data instead. 10000 simulated TACs were generated from the S3 Input Function (section 7.1.1) for k_2=k_3=0.1 and evaluated against all 24 reference curves of Table 11.1 (Fixed C_r-Simulation, see section 7.2.3). The results are samples of 10000 k_3, whose mean values are listed in columns 2 and 3 of **Table 11.2**, and which now exclusively reflect properties of the reference curves. Columns 2 and 3 show variations ranging from 0.886 to 0.1306 for Putamen and from 0.0843 to 0.1250 for Cerebellum, providing evidence that reference curves can indeed show tendencies to favor high or low k_3. The observed variations should reflect different Blood Input Functions of the subjects. If that is true, there ought to be significant correlations between Putamen and Cerebellum columns. This is indeed the case, we find 'corr'=0.8335. But we also find 'dist'=0.0076 which is quite large for columns that ought to be identical.

The P/C pattern arising from the differences is identical with Table 11.1. So this pattern does not depend on whether it is obtained from regional results of a cortex region, of Hippocampus or from simulations[1]. Hence we

[1] instead of comparing P/C patterns, one might prefer considering correlations between the differences of Putamen and Cerebellum columns. Comparing them in all possible ways gives 'corr'=0.9022, 0.9672 and 0.8504.

patient	Putamen	Cerebellum	difference in % of Put.		normalized difference
M01001	0.1010	0.0893	-11.6	P	2.037
M01002	0.0966	0.0921	-4.7	P	0.784
M01003	0.1151	0.1250	+8.6	C	-1.724
M01004	0.0886	0.0843	-4.9	P	0.749
M01005	0.1306	0.1204	-8.2	P	1.776
M01007	0.0960	0.0885	-7.8	P	1.306
M01008	0.0941	0.0989	5.1	C	-0.836
M01009	0.1058	0.1169	10.5	C	-1.933
M01010	0.0965	0.0991	2.7	C	-0.453
M01012	0.0986	0.0947	-4.0	P	0.679
M01013	0.0921	0.0876	-4.9	P	0.784
M01014	0.1052	0.0969	-7.9	P	1.445

Table 11.2: *Comparing reference curves based on simulations*

have evidence that

1. reference curves have inherent tendencies to favor higher or lower k_3 results
2. they partly reflect differences in the patient's Blood Input Functions
3. they also contain an artefact component tied to the reference masks themselves. It is reproducible across different sets of target curves, both simulated and measured.

11.2 Conclusion

The artefact component is a serious problem since it puts the validity of regional results in question. Since there is only one copy of C_r being used for computation of every parametric k_3 image, nothing will compensate for errors induced by that curve. This is unlike the effects of noise of C_t, which average out across a region. If noise of C_r leads to elevated or reduced k_3 values, it will do so on all voxels simultaneously, fully affecting regional results. Investigations into this type of error can only be attempted at sample level, but even so we are somewhat short of data, having only one pair of reference curves (from Putamen and Cerebellum) available per patient.

A major effort to obtain more information and use it to control and assess the effect of the artefact component, will be made in chapter 14.

Chapter 12

Comparison with COLOGNE

As long as COLOGNE was the only method available, we could do little more than consistency checks to validate it. The images should stay the same when modifying certain parameters of the method. One example is changing the reference region, see the previous chapter. Another is leaving out single frames from evaluation.

12.1 The skip2 Problem

There should be no substantial changes as a result of excluding frames. Every frame contributes a point to the value table of the TACs C_t. The identity of a TAC should not depend on one or two values. Consequently, q_1, k_2, k_3 computed from C_t should not be much affected.

Using the COLOGNE method for evaluation, we ran image-image comparisons after removing frames. At first, there were no significant breeches in consistency. But once the Zuendorf Mask had been removed, an area of the brain was revealed that behaved inconsistently. It contained the outward border of the Occipital Cortex in the median plane, an area known for its concentration of venous blood vessels [13].

If evaluation was carried out using all frames ("**skip0 policy**"), k_3 turned out significantly higher than if a **skip2 policy** (excluding frames 1 and 2) was adopted. The effect was clearly visible when comparing images.

From the beginning we suspected that blood volume effects were to blame. The kinetic model is based on the simplification of ignoring what ^{11}C signal is emitted directly from the blood (see Chapter 3, Assumption 1). Since radioactivity declines rapidly in the plasma after injection of the tracer, there should be little error inflicted by ignoring the blood signal in all but the very first frames.

But now we are faced with two conflicting k_3 readings from one region. So we must decide in favor of one or the other result and evaluation policy.

It might be helpful to have a “second opinion” available. In fact, it is this problem that made us want to implement NLS in the first place.

12.1.1 Quantifying the Problem

As mentioned previously, the phenomenon was first observed when comparing two images, computed with skip0 and skip2 policies. Later it was seen again on single images, of the $noise_{rel}$ modality (section 4.7.2). They had elevated intensity in the same region (Figure 12.1, top row). As explained in sections 4.7.2 and 4.7.3, this either points to noise or bias, and a decision may be reached by checking the *framedev* modalities. While other *framedev* images showed no unusual signs, on $framedev_2$ images the suspicious regions appeared in bright red. So the intensity of frame 2 was elevated compared to the Model Curve of the computed triple of constants.

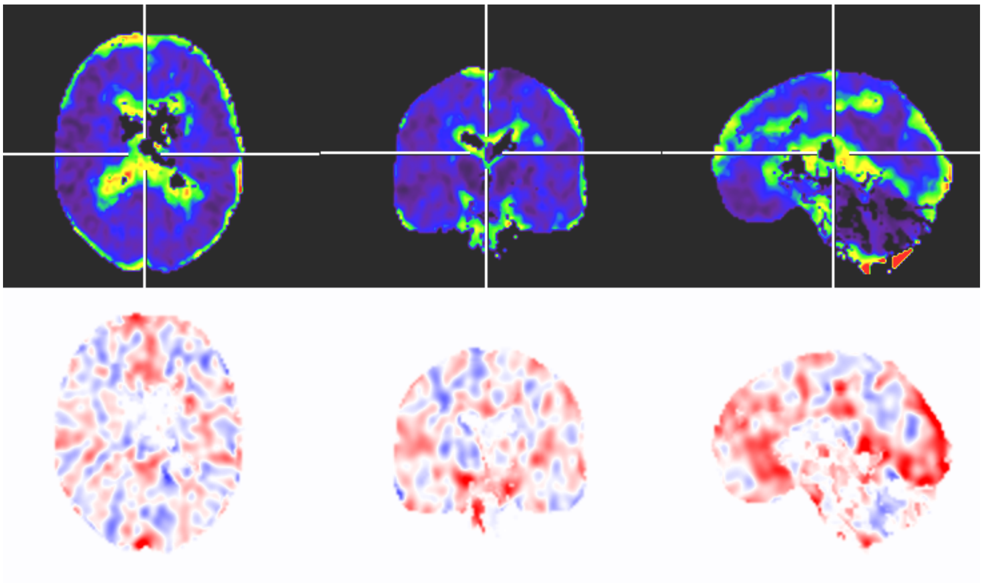


Figure 12.1: *Parametric images of* $noise_{rel}$ *(above) and* framedev2 *(below). Bright regions in the upper image are indicative of noise or bias, while deep red in the lower image is specific for the skip2 problem.*

In the biological context this finding makes a lot of sense. Frame 2 corresponds to the interval of 30 to 60 seconds after tracer injection. During this period, the injected bolus passes through the venous system for the first time, leading to a significant signal from the blood. It adds to the signal from the tissue, and the resulting TAC is no longer Model Compliant.

We created a mask from the 10690 (5%) voxels that had the highest $framedev_2$ readings. On this region, we subtracted the skip2 k_3 values from the skip0 values. The result averaged 0.044 if the COLOGNE method was used, and 0.015 if NLS was used. There were only 8 and 0, resp., voxels where the difference was negative. When restricting the mask to 6276 voxels, by selecting for low *k3dev* in order to remove outliers, there was still an average difference of 0.01953 (COLOGNE) and 0.00894 (NLS).

These results were obtained from a single PET scan, but they are representative of the rest. They show that there is a region of significant bias, and that the COLOGNE method is more vulnerable to the problem than NLS.

12.1.2 Deciding between the Policies

Another way to look at the data is by comparing whole images. We preselected for voxels that evaluated successfully on all 4 setups (skip 0/2 combined with COLOGNE/NLS). There was still a large number of noisy outliers, so we intersected with the Privileged Area of section 10.5.1, retaining the 66.88 percent of the brain that can be expected to evaluate robustly. On that set, pairs of images were compared using the ‘corr’ and ‘dist’ measures of section 2.8. The results are shown in Figure 12.2. Observations:

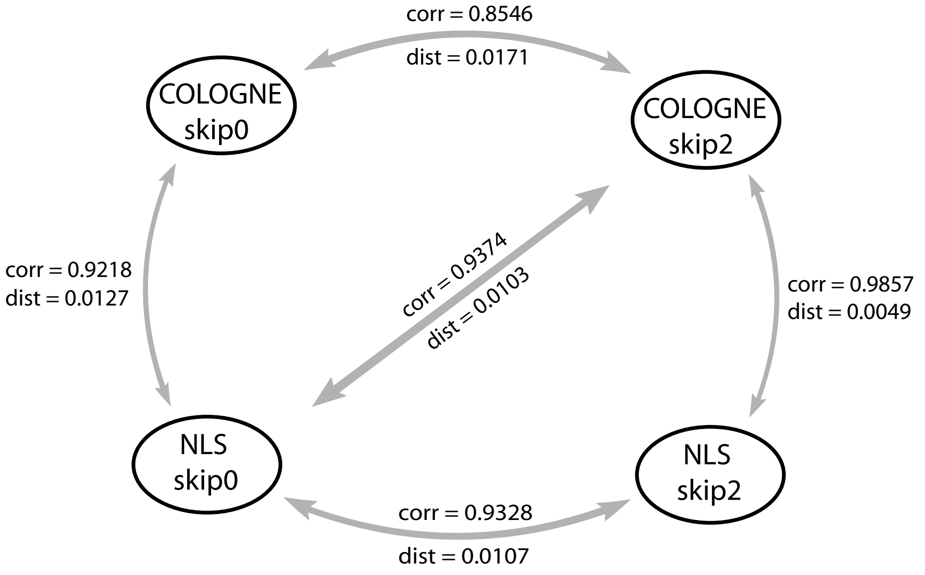


Figure 12.2: *Similarity measures between k_3 images obtained with different policies, restricted to the Privileged Area*

1. Comparing skip 0/2 COLOGNE images, we find low correlation and high differences. This can be looked upon as a restatement of the problem.

2. The same comparison on NLS images finds better scores on both measures. Apparently, NLS is more robust with respect to the problem,

as was already found in the previous section.

3. When computed with skip0, COLOGNE and NLS images show a poor match. It improves dramatically when both are computed with skip2.

4. If NLS with skip0 was to act as a referee between the COLOGNE images, it would favor the skip2 version.

12.1.3 Discussion

The most striking observation is to see the problem disappear if the first two frames are excluded from evaluation. This corroborates what has already been stated, namely, that it is caused by frame 2. This in turn supports the theory that we are looking at a blood volume effect, causing the TACs of certain voxels to be noncompliant with the kinetic model. So we are looking at bias rather than noise.

The second observation is that NLS gets derailed by this bias to a lesser degree than the COLOGNE method. The reasons will be discussed in section 13.4.

Together, these findings make NLS appear more trustworthy than the COLOGNE method, and the skip2 results more trustworthy than the skip0 results. This is supported by observation 4.

Presented data are taken from the PET scan of subject M01007. Such results are only obtained after sufficient preselection for voxels of good quality. If the selection was performed by *k3dev* thresholding, they were even more conclusive than with selection for membership in the Privileged Area. Without any selection, results were affected by noisy outliers, making them less conclusive, inconclusive or sometimes even contradicting the results presented here.

A skip2 policy has also been followed by G.Zuendorf in the study of 2002/2003 [14]. Back then, the first two frames were already perceived as a source of problems [45].

12.2 More Zonal Results

In chapter 10 we showed zonal statistics with image preprocessing in mind, leading to the policy of 'margin'=1 coupled with FWHM=8 which has been followed in the rest of this work. Now we present more zonal data aiming at validation of kinetic analysis. Again the subject is M01007 and statistics is restricted to the Privileged Area of section 10.5.1 in order to exclude noisy outliers. The zones are the same as in section 10.6.1, and q_1, k_2, k_3 have been computed with NLS using a skip0 policy.

	Number of Voxels:			Intensity				
	Total	Privil.	Privil.	of dec. w.				
zone	Brain	Area	Inters.	Sum	*simcr*	q_1	k_2	k_3
1	7600	1975	543	6476	.9523	.1884	.1339	.0963
2	6384	2290	1319	5262	.9552	.2521	.1132	.0763
3	10133	4507	3710	6576	.9600	.3096	.1046	.0702
4	15453	8330	7689	7941	.9651	.3704	.0912	.0673
5	23205	15333	14929	9832	.9671	.4645	.0927	.0692
6	40111	31443	31054	12073	.9677	.5814	.0976	.0729
7	64918	57068	56811	14266	.9691	.6896	.1011	.0775
8	49054	43109	42866	16354	.9708	.7893	.1045	.0827
9	11098	5793	5488	18582	.9812	.8048	.1031	.1169
10	7454	1487	1274	21951	.9933	.8089	.1145	.1757
11	6982	496	405	26128	.9970	.8584	.0691	.1854
12	6713	96	55	30911	.9987	.8842	.0158	.0963
13	2356	0	0					

Table 12.1: *Zonal mean values of the kinetic constants and simcr*

12.2.1 Kinetic Constants and *simcr*

Column 2 of **Table 12.1** shows again the distribution of the brain to the zones (The same data in percent is given in Table 10.3). Columns 3 and 4 list how many of these are in the Privileged Area and the Privileged Intersection Set of section 10.6. The right part of the table refers to only this subset.

Columns 7 to 9 show the zonal mean values of the kinetic constants. q_1 increases steadily with the zone numeration. This is parallel to the increase of intensity of the Decay Weighted sum (column 5), which is the quantity defining the zones. Since q_1 scales proportional to the Overall Intensity by Theorem 4.3, its behavior is in line with the expectations. Zone 12 overlaps with the reference regions. As this zone is approached, we see *simcr* climb towards 1, indicating increasing similarity of the TACs with C_r. We also observe an increase of k_3, which is known to have its maximum in the reference regions. k_2 stays close to 0.1 through most of the zones.

12.2.2 Limitations of NLS

Zones 11 and 12 show interesting outliers both of k_2 and k_3. Although k_3 is known to be high in the reference regions, the table shows a drop in zone 12, accompanied by a dramatic decrease of k_2. Considering *simcr* which is almost 1, the phenomenon can be explained with closeness of the TACs to the singularity of $\mathcal{M}$ (section 4.6.2): when TACs are similar to C_r, there is no way to tell if it is due to low k_2 or to high k_3. Apparently NLS with its

zone	yield[%]	Intensity	$noise_{abs}$	$noise_{rel}$	*iter*	*k3dev*	*uff*
1	27.5	6476	1028	0.2174	12.8	0.0955	0.2027
2	57.6	5262	1094	0.1782	12.1	0.0511	0.2968
3	82.3	6576	1049	0.1389	11.4	0.0306	0.3730
4	92.3	7941	997	0.1075	10.2	0.0219	0.4528
5	97.4	9832	961	0.0836	9.1	0.0186	0.5198
6	98.8	12073	899	0.0636	7.8	0.0152	0.5818
7	99.5	14266	886	0.0530	7.1	0.0121	0.6346
8	99.4	16354	887	0.0463	6.8	0.0151	0.6785
9	94.7	18582	964	0.0445	8.1	0.0471	0.6856
10	85.7	21951	996	0.0391	9.7	0.1092	0.7146
11	81.7	26128	940	0.0311	10.8	0.1465	0.7481
12	57.3	30911	874	0.0244	11.9	0.3462	0.7785

Table 12.2: *Markers relating to the quality of kinetic analysis*

current setup favors the local optimum which has k_2 an order of magnitude below the expected value of 0.1.

NLS was far less sensitive to the problem when we tried to reproduce it in the simulator (data unlisted). So the phenomenon might also be due to biased input data. For an overview of three types of bias that could be identified in real images, see section 15.10.

Table 12.2 shows the zonal behavior of different quality markers. *k3dev* goes up at both ends of the zonal scale, indicating results of poor precision. This is accompanied by loss of voxels during kinetic analysis (column 2). An additional witness to the difficulty of k_3 computation is the *iter* column, it shows the average number of iterations needed by NLS to process a voxel[1].

In the upper zones, the phenomenon can be explained by high *simcr* as in the previous paragraph. In the lower zones we need a different explanation. The $noise_{abs}$ marker has little overall variation. Therefore, $noise_{rel}$ comes out roughly inversely proportional to the Overall Intensity (column 3). Since $noise_{rel}$ is the inverse Signal-to-Noise Ratio (section 4.7.2), we have poor SNR resulting from poor signal strength in the lowest zones, explaining poor performance of kinetic analysis.

12.2.3 Comparing Results of NLS and COLOGNE

By ***k3diff*** we denote the voxel based difference between k_3 computed by COLOGNE and NLS, in this order. | *k3diff* | is its absolute value. In **Table 12.3**, both are averaged across the zones. *k3diff* columns show that COLOGNE is biased toward slightly higher results. | *k3diff* | columns re-

[1] the maximum number allowed is 20, see Chapter 5

	skip0		skip2	
zone	*k3diff*	|*k3diff*|	*k3diff*	|*k3diff*|
1	.0111	.0171	.0005	.0073
2	.0070	.0109	.0015	.0038
3	.0040	.0066	.0011	.0024
4	.0022	.0043	.0009	.0018
5	.0013	.0033	.0005	.0012
6	.0004	.0025	.0004	.0009
7	.0003	.0024	.0004	.0008
8	.0005	.0025	.0005	.0008
9	.0022	.0071	.0017	.0025
10	.0020	.0100	.0024	.0040
11	.0034	.0136	.0051	.0068
12	-.0108	.0147	.0014	.0094

Table 12.3: *Comparing k_3 of COLOGNE and NLS*

flect the average voxel based differences between k_3 computed with either method. Much of it averages out across the zones, as can be seen by comparison with *k3diff*. Further observations:

- By far the lowest differences are seen in the Medium Zones. | *k3diff* | deteriorates on both ends of the zonal scale, just as we have observed for *k3dev*, *iter* and yield in Table 12.2.
- absolute differences go down when switching to a skip2 policy, which is quite in line with the results of Figure 12.2.
- In the lowest zones, COLOGNE is strongly biased toward higher k_3, this almost completely disappears when switching from skip0 to skip2 policies. So the blood volume effect identified in section 12.1.3 emphasizes the low intensity zones.

Model curves of COLOGNE results showed residual squares about 1 percent above their NLS counterparts, which supposedly represent the possible minimum (data unlisted). So COLOGNE is remarkably successful as a minimizer of Decay Weighted squares. Such performance was only reached when running COLOGNE with Combined Weighting (section 7.3), while pure C_r- or Decay Weighting performed much worse. Further improvement, down to almost 1 permille of excessive squares, was observed after switching to the skip2 policy.

Chapter 13

Regional Results

Dissecting the brain into intensity zones as in the previous chapters is just a technical measure. The real interest will be on anatomic regions. This chapter presents an overview of typical results, allowing to localize some phenomena observed in the zonal statistics of chapter 12.

The main distinction in brain tissue is between "White Matter", consisting mainly of axons, and "Gray Matter", which also contains the neural cell bodies. In our atlas (section 10.1) White Matter, although it is spread across the whole brain, is treated as a single region. Of Gray Matter, the main distinction is between Neocortex and phylogenetically older areas. Neocortex happens to mostly coincide with the "Medium Zones" of chapters 10 and 12, which produce the most reliable k_3 results. It is subdivided in Frontal, Parietal, Temporal and Occipital Cortex. Two large regions defying this classification are Cingulate and Insular Cortex. Of the older brain areas, Cerebellum is the largest distinct entity, and much of the rest, including Putamen, is subsumed under the term "Basal Ganglia".

13.1 Kinetic Constants in Literature

In [17, 31], $k_3 = 0.07min^{-1}$ is reported for Parietal Cortex. According to [33] and based on Human post-mortem data, AChE activity in Caudate[1] and Putamen is 53-62 times that in Parietal Cortex - leading to an estimated k_3 between 3.7 and 4.3 min^{-1}. According to other post-mortem studies [1, 3], AChE activity in Cortex/Thalamus/Cerebellum/Striatum[2] relates as 1/3/8/38. With cortical $k_3 = 0.08$ as found in this work, this gives k_3=3 in Putamen and 0.65 in Cerebellum.

According to [17, 31] little variation of k_2 is expected in gray matter structures, the values ranging from 0.08 to 0.14 min^{-1}. This finding is linked by the authors to high diffusibility of MP4A across the Blood Brain

[1] a small region located close to Putamen

[2] a larger region containing the Putamen

Name	Voxels	Yield [%]	q_1	k_2	k_3	*k3dev*
White Matter	54494	95.76	.6622	.0817	.0728	.0384
Frontal Cortex:						
Superior Frontal Gyrus	14384	99.52	.7941	.1064	.0856	.0162
Middle Frontal Gyrus	7411	98.86	.8310	.1092	.0798	.0181
Inferior Frontal Gyrus	6014	98.06	.7976	.1083	.0839	.0198
Orbito Lateral Gyrus	2237	99.79	.8349	.0992	.0782	.0127
Orbito Medial Gyrus	2828	98.86	.8153	.0902	.0861	.0219
Precentral Gyrus	6525	99.39	.8326	.1121	.0968	.0179
Gyrus Rectus	1469	99.34	.8460	.0880	.0867	.0198
total Frontal	40868	99.13	.8130	.1059	.0858	.0178
Parietal Cortex:						
Superior Parietal Gyrus	4961	99.69	.7971	.1001	.0714	.0122
Supramarginal Gyrus	5751	99.75	.8244	.0983	.0730	.0120
Inferior Parietal Lobule	3798	99.70	.8113	.0995	.0655	.0104
Postcentral Gyrus	7315	99.31	.7937	.1088	.0871	.0162
total Parietal	21825	99.53	.8057	.1024	.0760	.0133
Temporal Cortex:						
Superior Temporal Gyrus	8516	99.20	.8023	.0987	.0812	.0179
Middle Temporal Gyrus	6540	99.26	.7688	.0880	.0712	.0145
Inferior Temporal Gyrus	4349	99.22	.7215	.0841	.0689	.0159
Fusiform Gyrus	4428	99.20	.7685	.0768	.0866	.0262
total Temporal	23833	99.22	.7721	.0890	.0772	.0186
Occipital Cortex:						
Superior Occipital Gyrus	2042	99.30	.7792	.1124	.0667	.0148
Middle Occipital Gyrus	2261	99.04	.7757	.1052	.0654	.0124
Inferior Occipital Gyrus	3163	99.10	.7997	.1112	.0671	.0154
Lingual Gyrus	3927	99.65	.9366	.1104	.0743	.0142
Cuneus	2378	99.64	.9660	.1199	.0641	.0091
Precuneus	3976	99.89	.9414	.1081	.0656	.0090
total Occipital	17747	99.49	.8789	.1109	.0677	.0127
Other:						
Insular Cortex	2505	99.01	.8406	.0908	.1044	.0245
Cingulate Gyrus	6315	99.19	.8534	.0957	.0846	.0183
Hippocampus	2520	97.13	.7561	.0895	.1112	.0406
Amygdala	591	88.10	.6668	.0994	.1558	.0905
Brain Stem	1317	69.48	.6889	.1078	.1721	.1710
total cortex	113093	99.29	.8162	.1016	.0796	.0167

Table 13.1: *Regional results: kinetic constants and k3dev. Computed using NLS with skip2 policy and a 16 ml Putamen reference mask, FWHM=8mm and 'margin'=1*

Barrier and the associated correlation of k_1 and k_2 with cerebral blood flow [15]. They go on to claim that because blood flow is similar in Caudate, Putamen and cortex, Putamen k_2 values are expected in the same range, i.e. around 0.1 min^{-1}.

Based on evidence of his own and of [19], Koeppe claims that the k_1/k_2 relation for MP4A and MP4P is rather uniform across the brain [23]. If that is true, we should expect little variation of k_1 as well.

13.2 Kinetic Constants and *k3dev*

k_3 in the Basal Ganglia is high and therefore notoriously hard to evaluate; one important exception is Hippocampus. Much of the rest consists of small nuclei which are hardly resolved in dynamic PET studies, given the fact that we need to use an 8mm Gauss Filter. So we leave them out of consideration in this chapter, and include Amygdala and Brainstem just for demonstration.

Table 13.1 contains regional results averaged over the subjects of Sample 1. Columns 3 and 4 give every region's number of voxels and which percentage of them survived failselection. The remaining columns refer to this subset.

Quite uniform behavior is seen in cortex regions: with failrates typically below 1 percent, k_3 is low and k_2 averages around 0.1. *k3dev* is as low as in the "Medium Zones" of the previous chapters. As we move toward the older brain, *k3dev* quickly deteriorates while k_3 rises, k_2 stays put and q_1 declines a little. The succession Hippocampus, Amygdala, Brainstem shows this development. The lowest q_1 is seen in White Matter. On images such as Figure 13.1, top row, q_1 was found to be as low as 0.3 in Corpus Callosum. Such values are not seen in the table because it doesn't subdivide the White Matter area.

Among cortical regions, Fusiform Gyrus and Insular Cortex have the lowest precision (*k3dev*=0.0262) and Cuneus, Precuneus the highest (0.0090). Interestingly, this is reflected by the k_2/k_3 relation which is lowest in the former and highest in the latter regions. Since high k_2 and low k_3 both facilitate k_3 computation (Table 7.3), this nicely explains the phenomenon.

13.3 Other regional Results

Table 13.2 has additional modalities of Sample 1 data. Its columns 2 and 3 show signed and unsigned differences of k_3 computed by COLOGNE and NLS. While column 3 gives an impression of the absolute voxel based differences, in column 2 they are allowed to average out, leading to regional differences. Even so, COLOGNE is biased toward higher values of k_3, an effect which is almost negligible in Neocortex but becomes more pronounced in hotter areas such as Hippocampus and Amygdala.

Columns 4 and 5 show regional mean values of *iter*, which is the number of iterations it took NLS to converge, and *simcr*, an indicator of similarity to the reference curve. Parametric images of both are in **Figure 13.2**. By definition, the reference regions are the hottest on the *simcr* image. On the *iter* image, we find parallels to the *k3dev* image of Figure 13.1. High values

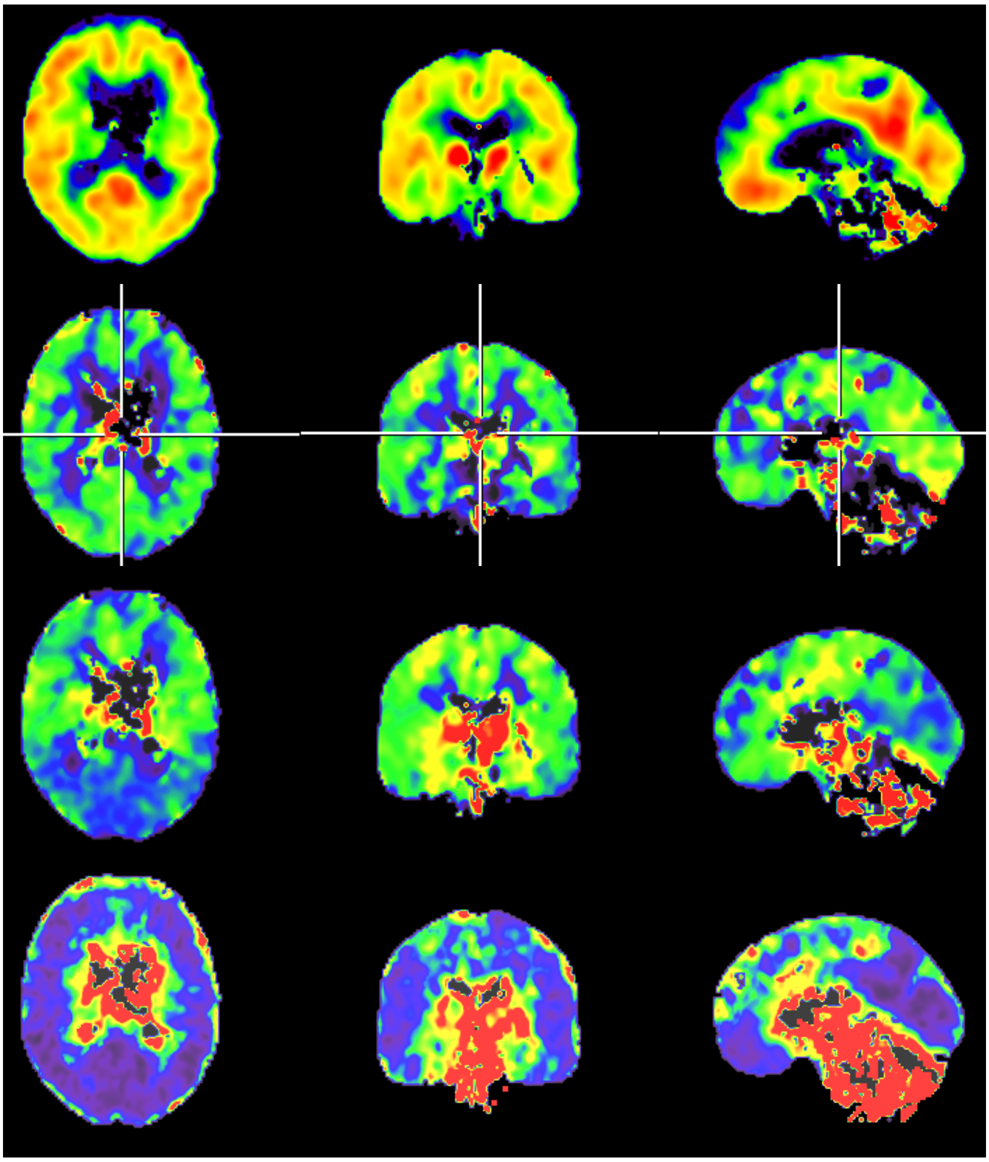


Figure 13.1: *Parametric images of* q_1*,* k_2*,* k_3 *and* k3dev *(from top to bottom) using a "rainbow" color scheme that maps increasing intensities to blue<green<yellow<red. Black areas refer to voxels whose evaluation failed.*

indicate poor convergence properties, which are either due to high *simcr* like in the reference regions, or poor signal strength like at the periphery of the brain and in the ventricles.

Name	*k3diff*	*|k3diff|*	*iter*	*simcr*
White Matter	.0007	.0018	8.99	.9718
Superior Frontal Gyrus	.0008	.0014	8.22	.9722
Middle Frontal Gyrus	.0005	.0012	7.83	.9690
Inferior Frontal Gyrus	.0006	.0013	8.13	.9714
Orbito Lateral Gyrus	.0003	.0010	7.79	.9712
Orbito Medial Gyrus	.0003	.0014	8.28	.9770
Precentral Gyrus	.0007	.0014	7.97	.9765
Gyrus Rectus	.0003	.0012	8.24	.9785
total Frontal	.0006	.0013	8.08	.9727
Superior Parietal Gyrus	.0005	.0010	7.72	.9657
Supramarginal Gyrus	.0003	.0008	7.48	.9681
Inferior Parietal Lobule	.0002	.0007	7.39	.9619
Postcentral Gyrus	.0006	.0013	7.97	.9730
total Parietal	.0004	.0010	7.68	.9681
Superior Temporal Gyrus	.0007	.0013	7.98	.9709
Middle Temporal Gyrus	.0003	.0010	7.93	.9695
Inferior Temporal Gyrus	.0002	.0010	8.43	.9682
Fusiform Gyrus	.0007	.0018	9.13	.9803
total Temporal	.0005	.0012	8.26	.9718
Superior Occipital Gyrus	.0005	.0009	7.64	.9559
Middle Occipital Gyrus	.0001	.0007	7.54	.9576
Inferior Occipital Gyrus	.0005	.0010	8.05	.9526
Lingual Gyrus	.0007	.0011	7.74	.9594
Cuneus	.0003	.0006	7.10	.9490
Precuneus	.0003	.0007	7.34	.9574
total Occipital	.0004	.0009	7.58	.9557
Insular Cortex	.0007	.0019	8.62	.9833
Cingulate Gyrus	.0005	.0013	8.22	.9746
Hippocampus	.0020	.0034	9.81	.9826
Amygdala	.0090	.0114	12.39	.9854
Brain Stem	.0093	.0139	13.19	.9925
total cortex	.0005	.0012	7.98	.9693

Table 13.2: *More columns of Table 13.1, computed with identical settings*

13.4 The skip2 Problem revisited

Columns 4 and 5 of **Table 13.3** have been computed using all frames ("skip0 policy"). They were obtained from the PET data of subject M01007. For the lower section of the table, we've selected the top scoring regions of **skip0-2**, which is the difference of k_3 computed with skip0 and skip2 policies. Their

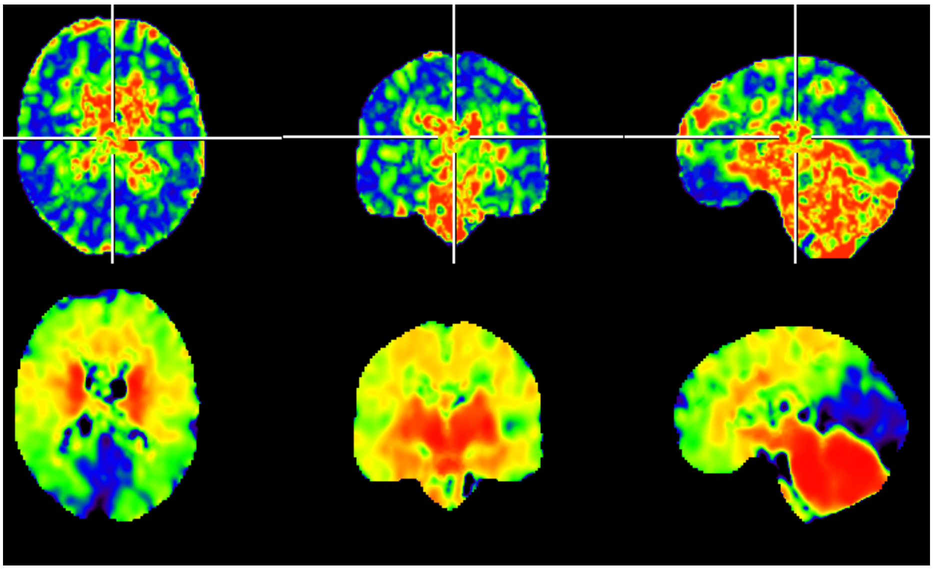


Figure 13.2: *Parametric images of iter (top row) and simcr (bottom row)*

scores are in column 2 for NLS and column 3 for COLOGNE. In the upper section, we have 3 “normal” regions for comparison.

Name	skip0-2 (NLS)	skip0-2 (COLOGNE)	*framedev2* (NLS)	*k3diff* (skip0)
Superior Frontal	.0017	.0022	551	.0011
Middle Frontal	-.0005	-.0018	-333	-.0011
Inferior Frontal	.0009	.0023	287	.0018
Orbito Medial	.0019	.0047	625	.0032
Superior Temporal	.0030	.0066	1127	.0041
Inferior Occipital	.0035	.0046	1758	.0015
Insular	.0032	.0076	1318	.0052
Cingulate	.0036	.0080	1276	.0052
Hippocampus	.0051	.0112	1207	.0084
Amygdala	.0210	.0903	3196	.0724
Brainstem	.0238	.0377	1898	.0019

Table 13.3: *Regional skip2 effects*

Column 3 dominates over column 2, showing once again that COLOGNE is more affected by the problem than NLS. While having been selected for their skip0-2 differences, the regions of the lower section also contain the 7 highest *framedev2* readings of the 26 regions[3] of Table 13.1, and the 5

[3] data of other regions not listed

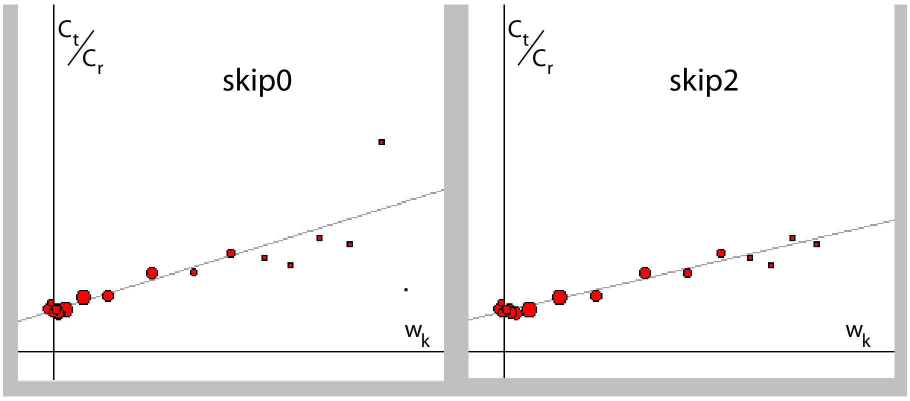


Figure 13.3: *Response of COLOGNE to outlying early frames, with skip0 and skip2 policies. The dots correspond to the frames, their sizes to the weighting (VINCI MP4A plugin, screenshots of the COLOGNE testbed).*

highest resulting *k3diff*. Bias caused by blood volume effects is therefore a major reason for COLOGNE overestimating k_3.

Why would COLOGNE be more sensitive toward this kind of bias? Recall that it finds the k that maximizes the correlation of a plot of $\frac{C_t}{C_r}$ against w_k. It then goes on to compute k_3 from the parameters of the regression line through that cloud of points. **Figure 13.3** illustrates how the regression line changes when two frames are removed. The cloud has been plotted (for its respective optimal k) with and without the two dots on the right side, which correspond to frames 1 and 2. The larger dot is frame 2. It pulls the regression line upwards, resulting in k_3=0.0747. Remove the two dots, and the slope declines, leading to k_3=0.0621. This example, taken from a voxel of the Occipital Cortex of M01007, is a moderate case, since frame 1 (the smaller dot) partially neutralizes the influence of frame 2. More extreme behavior was seen in cases where frame 1 pulled in the same direction. So we find that

- although frame 1 and 2 are suppressed by the weighting, they have an overproportional influence on the regression line, because of their peripheral position. In high k_3 areas their leverage is even greater, since the remaining points are huddled together in a cluster on the left side.

- C_r- and Decay Weighting both reduce the influence of frames 1 and 2: the frames are short so their Decay Weights are low, and C_r is low during the first minute. So the weightings have a beneficial influence, both with respect to this problem and in general.

So at the heart of COLOGNE being vulnerable to blood volume effects, we find the properties of correlation coefficient and of linear regression: both are notorious for their sensitivity to outliers. So it is a natural idea to look for a method with more “democratic” behavior toward the frames, and this is what turned our attention to NLS.

Chapter 14

Region based Error Estimates

k_3 is computed from pairs of TACs: C_r of the reference region, C_t of the target region. Noise of either part of this input may translate to random error or - via nonlinearity effects - bias of regional k_3 results. In this and the following chapter we assess separately all four of these contributions, based on data from PET images. In addition, we search for an optimal mask generating strategy to minimize error caused by C_r.

14.1 Error induced by C_r

C_r is obtained by sampling the unfiltered normalized frames using a reference mask, which is computed by the adaptive strategy of section 10.3. In chapter 11, we found evidence that regional k_3 results depend heavily on whether Putamen or Cerebellum is used as reference tissue. Since both are considered valid reference areas, we have no way of deciding which one leads to the "right" results. The situation is unsatisfying and warrants a detailed and systematic investigation.

Methods

14.1.1 Patients and Prerequisites

Investigations in this chapter are based on two samples of patients: **Sample 1** consists of 12 volunteers diagnosed with MCI who were scanned at MPIFNF in 2007, **Sample 2** contains 17 patients of different diagnostic groups, 2 normal controls, 4 diagnosed with MCI and 11 with AD. All had been scanned at MPIFNF in 2002 or 2003 using the "traditional" framing.

Reference masks were characterized by a precursor mask, a number of voxels (sized $2 \times 2 \times 2$ mm) and the 'padding' parameter of section 10.3.

Precursor masks for Putamen and Cerebellum were taken from the atlas of section 10.1 and had 1557 and 23229 voxels, respectively. Unilateral precursors were obtained by splitting the above in their left and right hemispheral parts. As usual, "Putamen 674+2" is the mask obtained from the Putamen precursor, setting the number of voxels to 674 (5.392 ml) and 'padding' to 2. We examined the following bilateral or **full masks**:

1. Putamen 674+2 (5.392 ml)
2. Putamen 1557+3 (12.456 ml)
3. Putamen 2000+3 (16 ml)
4. Putamen 4000+4 (32 ml)
5. Putamen 8000+5 (64 ml)
6. Cerebellum 4000-2 (32 ml)
7. Cerebellum 8000-2 (64 ml)

After the padding step, they had grown or shrunk to 3941, 5528, 5528, 7410, 9608, 12919 and 12919 voxels, respectively, from which the final selection was made. We also used left and right **unilateral masks**, obtained from aforementioned left and right precursors, using half the indicated number of voxels and identical padding. By construction, left and right masks did not overlap, ensuring stochastic independence of left and right reference curves.

For comparing vectors of results, we use the similarity measures **'corr'**, **'dist'** and **'shift'** of section 2.8.

14.1.2 k_3-Digests

In chapter 11, we found that the reference curves C_r were to blame for considerable differences of the computed k_3. It could be shown that they have a tendency to favor higher or lower k_3, by pushing all voxel based results into the same direction.

Encouraged by the results of Table 11.2, we selected k_3-Digests as a means of quantifying this up- or downshifting tendency. The **k_3-Digest** of a reference curve C_r is the mean of a large sample of k_3 values, which are the results of kinetic analysis of 10000 simulated TACs against the given C_r.

The rationale for adopting this approach is to achieve independence of the patient's target tissue. C_r is the only measured input, so the k_3-Digest is a property of C_r. One may ask if it should not be based on a single simulated TAC instead of a large sample. But we wish to include noise in the simulations, and a single TAC cannot properly represent all manifestations of noise.

Of course the digest depends on all parameters used both in the simulations (such as k_1,k_2,k_3, FWHM, the Blood Input Function) and kinetic analysis. We keep these parameters fixed to the values of section 7.2.4, except that we use a skip2 policy. The other exception is k_3, which is set to 0.0790 and 0.1231 in order to create two flavors: **Cortex Digests** and **Hippocampus Digests**.

The digest of a reference curve C_r reflects its tendency to produce higher or lower k_3 results. When viewed across a sample of patients, it shows considerable variation, as visible in the columns of Tables 11.2 or 14.3. Such variability is expected: recall that we evaluate C_t arising from the S3 Input Function of section 7.1.1 which remains constant, against C_r arising from Blood Input Functions that are different for every patient. These differences account for most of the variability seen in the digests.

But this is not the type of variability we are looking for. The real interest is in comparing reference curves of the same patient, but obtained with different techniques. As they should represent the same Blood Input Function, we expect no difference of their digests to begin with. Any differences observed will be effects of noise or bias of C_r.

14.1.3 "Second Opinion" Statistics

There are three places in this chapter where we have data samples of size 2, belonging to N different populations. The aim is to estimate SD of C_t-induced or C_r-induced random error, of k_3 results of a "typical" patient. Suppose, for a moment, we had two independent PET scans of one patient. This would give two independent reference curves and allow to study the difference between their digests. It relates to the dispersion of a fictive, large population of possible measurements that could be obtained from this patient, in the following way: looking upon two k_3-digests x and y as a sample of size n=2, picked from this population, and $\mu = \frac{x+y}{2}$ its mean value, we compute the sample variance by formula (2.2):

$$\frac{(x-\mu)^2 + (y-\mu)^2}{2-1} = \frac{(\frac{1}{2}(x-y))^2 + (\frac{1}{2}(y-x))^2}{1} = \frac{1}{2}(x-y)^2 \qquad (14.1)$$

It is an estimator of the population variance. Another patient represents a different population (his Blood Input Functions are different even if procedures remain unchanged). Now consider a patient sample of size N. If a patient is picked from this sample at equal probability and the variance of his k_3-digest is taken, then the expectancy of this random experiment is estimated by the mean of the N estimators of equation (14.1):

$$\frac{1}{2N} \cdot \sum_{i=1}^{N} (x_i - y_i)^2 \qquad (14.2)$$

It predicts the variance of k_3-digests of a typical patient of the sample, and is more reliable than a quantity based on only 2 measurements. The corresponding standard deviation happens to be a scalar multiple of the similarity measure 'dist' of section 2.8:

$$SD \approx \frac{1}{\sqrt{2}} \cdot `dist' = \sqrt{\frac{1}{2N} \cdot \sum_{i=1}^{N} (x_i - y_i)^2} \tag{14.3}$$

We therefore list 'dist' in the results section of this chapter, and later apply the factor $\frac{1}{\sqrt{2}}$ to convert to standard deviations.

14.1.4 abba-Splits

In reality, we do not have two acquisitions per patient. The next best option is to split one PET scan in two, which should be roughly equivalent and stochastically independent. The split can be done at the sinogram level, which is the stage before image reconstruction. Sinogram files can be viewed as huge histograms of coincidence events, containing the number of counts for each LOR (section 1.1) of the scanner in a given time interval.

Sinograms can be computed from listmode data if available. The listmode format contains an entry with a time stamp for every recorded coincidence. Given such data, one can distribute the information to two sets of sinograms, counting events that happend during even or odd seconds, respectively. Some scanners offer the option of "gating", where an external periodic trigger directs the events toward one or the other set of sinograms.

Unfortunately, the scanner we used had no support for listmode or gating. The next best option is to record as many timeframes as possible into separate sinograms, and recompose them, in different ways, to larger units before reconstruction. We recorded 136 initial sinograms (to be called **sub-sinograms**) and assembled the final framing of **Table 14.1**. Within each frame, data were splitted by assigning each sub-sinogram to one of 2 series, called a and b, as shown in **Figure 14.1**. The splitup pattern was **'abba'** when there were 4 sub-sinograms, **'ababbaba'** when there were 8. By adding up just the a-sinograms of every frame, we assembled a full PET acquistion (called **abba-a**) at sinogram level, consisting of all concidences counted during an a-subinterval. Likewise, we obtained an abba-b sinogram file. Both were scaled with a factor 2 to compensate for halved amounts of radioactivity. Third, an **abba-t** file (t="total") was obtained by adding up all sub-sinograms of every frame.

The files were reconstructed independently to make abba-a, b and t dynamic PET image series. The timing information supplied to the reconstruction software was that of Table 14.1 in all cases, so as to ensure identical Decay Correction. a- and b- sinograms can be viewed as representative subsets of all coincidences counted during every frame.

Frame nr.	Start Time [s]	Duration [s]	Number of Sub-Sinograms
1	0	20	4
2	20	20	4
3	40	20	4
4	60	20	4
5	80	20	4
6	100	20	4
7	120	20	4
8	140	40	8
9	180	40	4
10	220	40	4
11	260	40	4
12	300	60	4
13	360	120	8
14	480	120	8
15	600	120	4
16	720	240	8
17	960	240	8
18	1200	400	8
19	1600	400	8
20	2000	400	8
21	2400	400	8
22	2800	400	8
23	3200	400	8

Table 14.1: *The abba-t Framing and its composition of initially recorded sinograms*

Note that we kept the distribution schemes ‘abba’ and ‘ababbaba’ symmetric in order to avoid a temporal offset between the series, such as resulting from an alternating pattern ‘abab. . . ’. This makes it possible to evaluate (as in section 14.2) TACs from abba-a or b-images against reference curves sampled from abba-t images without inviting major bias. Some residual bias is still expected arising from the convexity of the Decay Function, leading to slightly reduced signal in the “inner” sinograms of the b-series, but this effect was considered negligible.

The sub-sinograms could also be grouped to make the “traditional” scheme of 20 frames (section 6.1.1). The corresponding images have been used in most of this work. However with the abba-framing, we had to break this tradition in order to achieve divisibility by 4, of the number of sub-sinograms per frame, and make the symmetric splitup patterns possible.

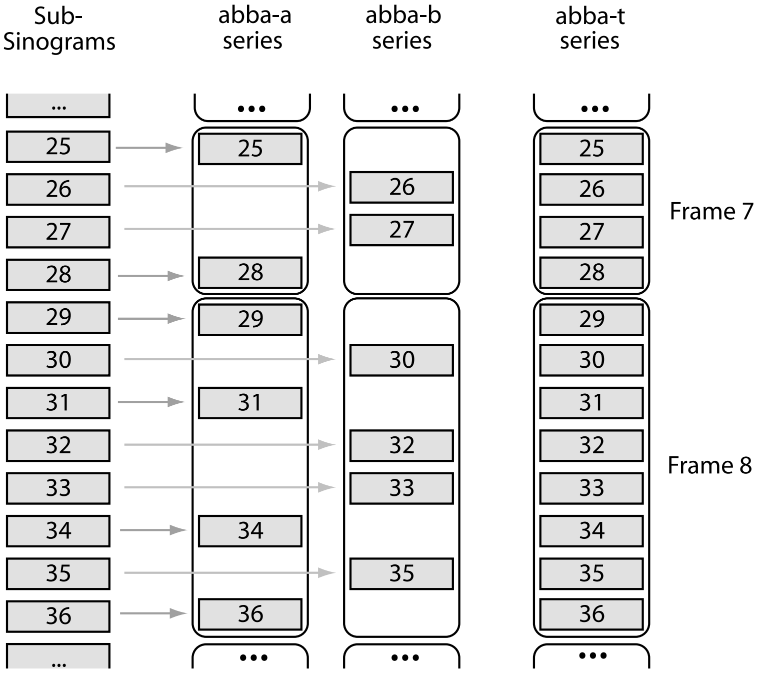


Figure 14.1: *Scheme illustrating how the sinogram frames of abba-a, b and -t series arise by summation of subsets of the sub-sinograms. Each sub-sinogram covers 5 seconds of scantime.*

14.1.5 L,R-Splits

Since abba-splits were available only for Sample 1 where we had used appropriate acquisition protocols, another method was employed for splitting radioactivity of existing images. If Putamen is supposed to be an ideal reference region, the same is obviously assumed for a patient's left and right Putamen. We therefore expect valid reference curves from **unilateral Putamen masks** as well, whose volume is half of the corresponding **full mask**, and whose 'padding' is identical. Like in the abba-splits the amount of contributing counts is cut in half, leading to more noise. Applicability is limited to investigations on C_r, since we cannot generalize the assumption of lateral symmetry to the target tissues.

14.1.6 Consistency Check: abba-t versus "Traditional"

abba-t and "traditional" image series both arise from full amounts of radioactivity, but are assembled via different framing schemes. Comparing them provides another "second opinion" effect, however stochastically dependent. For each patient of Sample 1 and reference curves obtained from either image, we took the Hippocampus Digest, and compared them (as

	full	left	right	full	left	right	full	left	right
voxels:	674	337	337	2000	1000	1000	4000	2000	2000
'corr'	.9485	.9466	.9658	.9972	.9747	.9955	.9985	.9916	.9972
'dist'	.0039	.0037	.0049	.0011	.0022	.0017	.0010	.0017	.0014
'shift'	.0005	.0011	.0014	-.0005	.0000	-.0007	-.0003	.0000	-.0004

Table 14.2: *Comparing abba-t and "traditional" results for Sample 1 and Putamen masks of different sizes*

vectors across Sample 1) using the similarity measures of section 2.8. The results, for different types of full and unilateral Putamen masks, are given in **Table 14.2**. We found a good match in every case, much closer than in the data shown later. Having seen Table 14.2, we decided to base all L,R-investigations on only one image series, the "traditional" one.

14.1.7 Properties of the Digest

14.1.7.1 Random Numbers

Dependency on the random numbers used in the simulations turned out to be low, affecting k_3-digests only in their fourth digit. Still we chose to eliminate this factor completely, by using the same set of 10000 TACs for all Hippocampus Digests and another fixed set of 10000 curves for all Cortex Digests.

14.1.7.2 Reproduction of k_3

Given the difference in Input Functions used for C_t and C_r, we cannot expect the digest to exactly reproduce k_3=0.1231 or 0.0790 which are used for the simulations. On the subject level, differences are huge for reasons given in section 14.1.2. On the sample level we found an average Hippocampus Digest of 0.1252 in Sample 1 and 0.1117 in Sample 2, for C_r obtained with 674+2 Putamen masks. Average Cortex Digests were 0.807 in Sample 1 and 0.750 in Sample 2. Note that

- even at sample level, differences can be expected as a result of different protocols followed by the teams who acquired the data (how fast is the tracer being injected, what is the total fluid volume)
- any bias we expect to measure for different types of masks is of course contained in the above figures.

Since we are interested in differences between digests rather than in their absolute values, we saw no reason to modify the Input Function and improve the match with Sample 2. It seemed more important to use the same measuring technique for reference curves of all samples and types of masks.

14.1.7.3 Translation to Regional Results

The digests have been adopted in order to exclude influence of the patients' target tissue. Yet on the sample level, their relation to regional results will be studied once, to obtain calibration factors for translating one situation to the other. We based this calibration on the series of reference curves that had the highest L,R-differences, obtained from unilateral 337+2 Putamen masks.

	Hippocampus Digests			Hippocampus regional		
				Mean Values		
Subject	left	right	left-right	left	right	left-right
M01001	0.1186	0.1460	-0.0274	0.0984	0.1262	-0.0278
M01002	0.1173	0.1422	-0.0249	0.0961	0.1197	-0.0236
M01003	0.1197	0.1490	-0.0293	0.0960	0.1253	-0.0293
M01004	0.1283	0.1009	0.0274	0.1427	0.1017	0.0410
M01005	0.1433	0.1591	-0.0158	0.1157	0.1257	-0.0100
M01007	0.1002	0.1309	-0.0307	0.0959	0.1364	-0.0405
M01008	0.1323	0.1078	0.0245	0.1343	0.1015	0.0328
M01009	0.1255	0.1152	0.0103	0.0836	0.0754	0.0082
M01010	0.1206	0.1073	0.0133	0.1103	0.0942	0.0161
M01012	0.1168	0.1207	-0.0039	0.1174	0.1244	-0.0070
M01013	0.1124	0.1144	-0.0020	0.1000	0.1034	-0.0034
M01014	0.1341	0.1315	0.0026	0.1152	0.1135	0.0017

Table 14.3: *Relation of Hippocampus Digests to regional Results (for Sample 1 and unilateral 337 voxel Putamen masks)*

In columns 2 and 3, **Table 14.3** shows Hippocampus Digests of reference curves from left and right masks, for all subjects of Sample 1. Columns 5 and 6 contain mean k_3 results of the Hippocampus region, sampled from parametric images that had been computed using these reference curves. Between corresponding columns we find low correlations and high distances: 'corr'=0.4945 and 'dist'=0.0200 between columns 2 and 5, 'corr'=0.6941 and 'dist'=0.0201 between columns 3 and 6. This is expected, since unlike columns 2 and 3, columns 5 and 6 involve every patient's Hippocampus tissue. So the Hippocampus pathology pattern across the patient sample weighs in and breaks the correlation. This changes when differences are considered as in columns 4 and 7. They mainly reflect properties of C_r, although modulated by the target tissue in column 7. We find 'corr'=0.9825 and 'dist'=0.0058 between columns 4 and 7.

As the data sets are well correlated, we can compare their amplitudes by computing the quotients

$$\frac{\sqrt{\sum_{i=1}^{N} {x_i}^2}}{\sqrt{\sum_{i=1}^{N} {y_i}^2}}$$

where x_i, y_i are the entries of columns 7 and 4 and N is the sample size. Likewise, we proceed for Sample 2 and repeat the same procedures for the Cortex Digests. We thus obtained the factors of **Table 14.4**.

	Average cortex	Hippocampus
Sample 1	1.1541	1.1800
Sample 2	1.0893	1.3221

Table 14.4: *Calibration factors used for computing regional random error and bias from k_3-digests*

14.1.8 Upscaling to full Amounts of Radioactivity

Reference curves used in L,R- or a,b-comparisons arise from halved amounts of radioactivity. We therefore need a way to extrapolate to the full amount. To investigate the response of k_3 digests to varying amounts of tracer, we conducted the following study based on C_r Block Simulations (section 7.2.2). 100 reference curves were simulated and their Hippocampus Digests computed[1]. Then we took their SD. Thus we proceed for 12 combinations of reference volume and tracer dosage, leading to **Table 14.5**. Since we expect them to be inversely proportional to the square root of counts involved in the data, we take all table entries times the square root of volume times bolus leading to **Table 14.6.** There we find almost constant rows, indicating that the response to bolus changes is as predicted. We therefore feel justified in using $\frac{1}{\sqrt{2}}$ as a conversion fator for extrapolating from 277.5 to 555 MBq.

Reference	Bolus[MBq]		
volume [ml]	277.5	555	1110
5.392	.012017	.008483	.005981
16	.007405	.005191	.003650
32	.005623	.003921	.002752
64	.004454	.003090	.002164

Table 14.5: *k_3 standard deviations of C_r Block Simulations*

[1] in this study, the digests were based on only 1000 simulated TACs

Reference	Bolus[MBq]		
volume [ml]	277.5	555	1110
5.392	0.4648	0.4641	0.4627
16	0.4943	0.4892	0.4864
32	0.5299	0.5226	0.5186
64	0.5935	0.5823	0.5766

Table 14.6: *establishing a conversion rule (see text)*

The columns are not constant, which is explained by the fact that increasing the reference volume reduces noise of C_r, but not of C_t. Recomputing the tables for Cortex Digests revealed similar behavior, although on a level of error reduced by a factor of 2.14.

A similar study (unlisted) was conducted to justify use of $\frac{1}{\sqrt{2}}$ as a conversion factor in section 14.2.

Results

14.1.9 Symmetric Comparisons

Left/right comparisons (**L,R-comparisons**) of unilateral masks of 7 types, 5 Putamen and 2 Cerebellum, were performed. We list the raw data for only one example: Putamen masks of 337 voxels or 2.696 ml, in columns 2 and 3 of Table 14.3. Small masks like these exhibit so much noise that the columns are almost uncorrelated ('corr'=0.1045) and separated by an average distance of 'dist'=0.0206. Their mean values (left minus right) are separated by 'shift'=-0.0047. 'dist' is the main result, we use it in chapter 15 to compute error estimates.

Such studies were conducted 4 times, using different series of images or changing between Hippocampus- and Cortex Digests. Results are collected in **Table 14.7**. Two further studies involved **a,b-comparisons**, where we used full masks and compared C_r sampled from abba-a images with C_r sampled from abba-b images. These results are in **Table 14.8**. Although being mainly interested in 'dist', we also list 'corr' and 'shift' (see section 2.8): 'corr' is another indicator of similarity; 'shift' is expected close to zero since there should be no bias in symmetric situations. We leave the discussion for chapter 15.

14.1.10 Cross Comparisons

between different types of full masks are shown in **Table 14.10**. Its upper triangle is based on Cortex Digests, its lower triangle on Hippocampus Digests. Here 'shift' is the main result, it serves to measure relative bias. Similar data for Sample 2 are listed in appendix A.6.

	Putamen					Cerebellum	
	2.696 ml	6.232 ml	8 ml	16 ml	32 ml	16 ml	32 ml
Sample 1	.1045	.4503	.6508	.9415	.9491	.8960	.8850
“traditional”	**.0206**	**.0128**	**.0105**	**.0063**	.0078	**.0068**	**.0068**
Hippo Digests	-.0047	-.0026	-.0015	-.0019	.0013	.0008	-.0001
Sample 1	-.0111	-	.7062	.9545	-	-	-
abba-t	.0201	-	.0104	.0063	-	-	-
Hippo Digests	-.0043	-	-.0021	-.0024	-	-	-
Sample 2	.6991	.8635	.8761	.8756	.7843	.8444	.9181
“traditional”	**.0100**	**.0070**	**.0061**	**.0058**	.0107	**.0065**	**.0052**
Hippo Digests	.0002	.0014	.0006	.0009	.0042	.0004	.0014
Sample 1	.1601	.4671	.6742	.9387	.9602	.8924	.9097
“traditional”	**.0092**	**.0057**	**.0047**	**.0030**	.0027	**.0033**	**.0031**
Cortex Digests	-.0009	-.0004	-.0002	-.0011	.0003	.0000	-.0002

Table 14.7: *L,R-comparison studies. Every block contains ‘corr’, ‘dist’ and ‘shift’ from top to bottom. Subtraction order for ‘shift’ is L-R. The main results (bold print) have been used to compute the corresponding rows of Table 15.1.*

14.1.11 Trying to prove Asymmetry in Putamen

In Table 14.7 we find large L,R-differences between the small Putamen masks of Sample 1, exceeding the corresponding a,b differences of Table 14.8. They correspond to the raw data shown in columns 2 and 3 of Table 14.3. Since we had reason to believe they might be caused by more than random effects, we designed the following test. Column 4 of Table 14.3, containing the L,R-differences of 2.696ml unilateral Putamen masks, was recomputed twice: for abba-a and abba-b images. If the discrepancies are caused by asymmetries in the patient’s Putamen, it should lead to correlation between the two data sets. We found ‘corr’=0.4749, the corresponding P-value in a sample of 12 is 0.059 (one-tailed Spearman test). So we end up close to the border of significance. The effect was not observed for larger Putamen masks (8ml), where the correlation dropped to 0.1534.

14.1.12 Comparing full Masks by other Criteria

Some more obvious properties should also be monitored before deciding which mask is best. “Intensity” was computed at the patient level by averaging over the “Late Frames” (corresponding to the time interval 10-60 minutes p.i.) of a reference curve. Results were then averaged over the patients of Sample 1. The maximum score of 37253 Bq/ml belonged to the

	Putamen				Cerebellum	
	5.392 ml	12.456 ml	16 ml	32 ml	32 ml	64 ml
Sample 1	.4550	.8169	.8499	.9400	.9528	.9428
Hippocampus	**.0129**	**.0078**	**.0070**	**.0058**	**.0046**	**.0047**
Digests	-.0060	-.0041	-.0029	-.0022	-.0012	-.0016
Sample 1	.5259	.7705	.8086	.9372	.9709	.9635
Cortex	**.0062**	**.0044**	**.0040**	**.0029**	**.0019**	**.0020**
Digests	-.0032	-.0024	-.0019	-.0016	-.0008	-.0008

Table 14.8: *a,b-comparison studies of full masks. The blocks contain 'corr', 'dist' and 'shift' from top to bottom. Subtraction order of 'shift' is a-b. The main results (bold print) have been used to compute the corresponding rows of Table 15.1.*

smallest Putamen mask. The rest are shown as fractions of it in column 2 of **Table 14.9**. For columns 3 to 6, k_3 and *k3dev* were averaged over total cortex and over Hippocampus, then over Sample 1. k_3 columns should confirm the bias that has been read from the digests, and so they do (except for the largest Putamen mask where the bias is lower than in the simulations). *k3dev* and the voxel yield (section 10.5) are listed as quality markers. We leave the discussion for chapter 15.

14.1.13 Visualization

Reference curves of 6 different types were averaged over the 12 subjects of Sample 1 to generate plots with error bars. To render the error bars meaningful, the curves had to be normalized to compensate for their different intensities. Individual curves were scaled such as to have the mean value of frames 11 to 20 equal 1. Then the curves were averaged across the sample and had their point-wise standard deviations computed by formula (2.2). Frames 11 to 20 thus ended up with smaller errors than frames 1 to 10, so

Type of Mask	Intensity	k_3 (regional)		*k3dev* (regional)		yield
		Hippo	Cortex	Hippo	Cortex	[%]
Put 5.392 ml	1.0000	.1107	.0795	.0401	.0170	89.22
Put 12.456 ml	.9142	.1118	.0800	.0400	.0168	89.00
Put 16 ml	.8744	.1112	.0796	.0406	.0167	88.72
Put 32 ml	.7419	.1113	.0786	.0554	.0189	86.04
Put 64 ml	.5941	.1144	.0783	.1920	.0270	81.27
Cbl 32 ml	.8686	.1095	.0788	.0453	.0184	87.12
Cbl 64 ml	.8107	.1077	.0780	.0465	.0175	86.96

Table 14.9: *Some other properties of mask types, averaged over Sample 1*

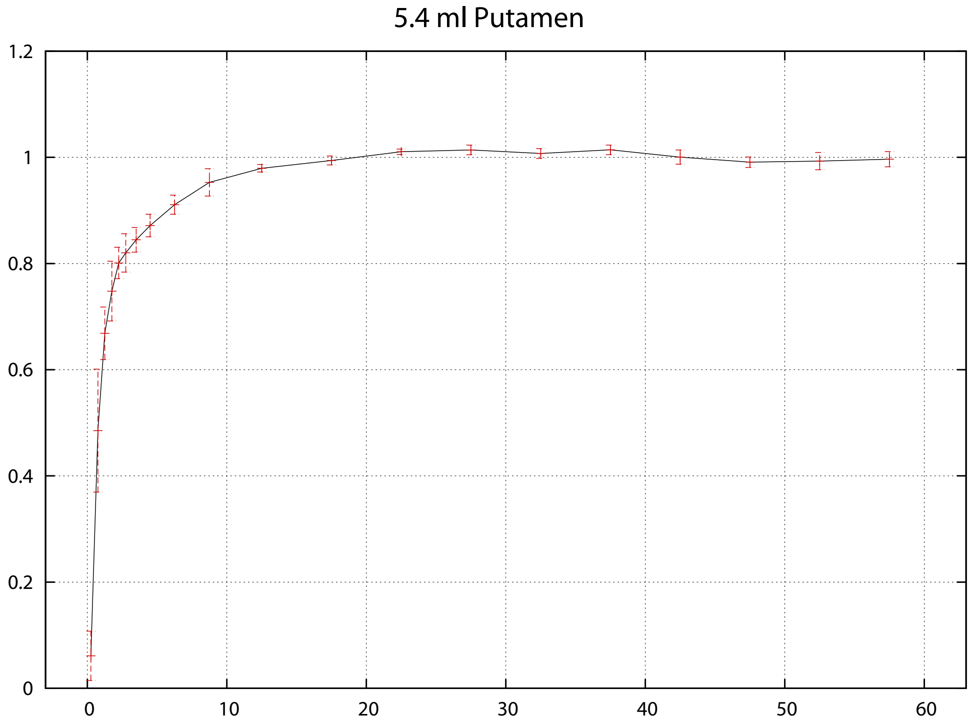
	Putamen				Cerebellum	
	5.392	16	32	64	32	64
	ml	ml	ml	ml	ml	ml
Putamen	-	.9023	.8900	.8508	.7129	.7671
5.392 ml	-	.0023	.0034	.0052	.0046	.0041
	-	-.0001	.0009	.0009	.0006	.0014
Putamen	.8881	-	.9628	.9432	.8302	.8348
16 ml	.0052	-	.0027	.0045	.0036	.0035
	.0007	-	.0009	.0015	.0006	.0015
Putamen	.8842	.9610	-	.9900	.8931	.8916
32 ml	.0075	.0054	-	.0021	.0031	.0032
	.0017	.0010	-	.0006	-.0003	.0006
Putamen	.8488	.9396	.9935	-	.8992	.8916
64 ml	.0178	.0160	.0116	-	.0041	.0044
	.0097	.0090	.0080	-	-.0009	.0000
Cerebellum	.6544	.7906	.8737	.8911	-	.9869
32 ml	.0103	.0082	.0076	.0163	-	.0015
	-.0006	-.0012	-.0023	-.0103	-	.0009
Cerebellum	.6986	.7786	.8605	.8732	.9849	-
64 ml	.0091	.0081	.0086	.0183	.0031	-
	-.0023	-.0029	-.0040	-.0120	-.0017	-

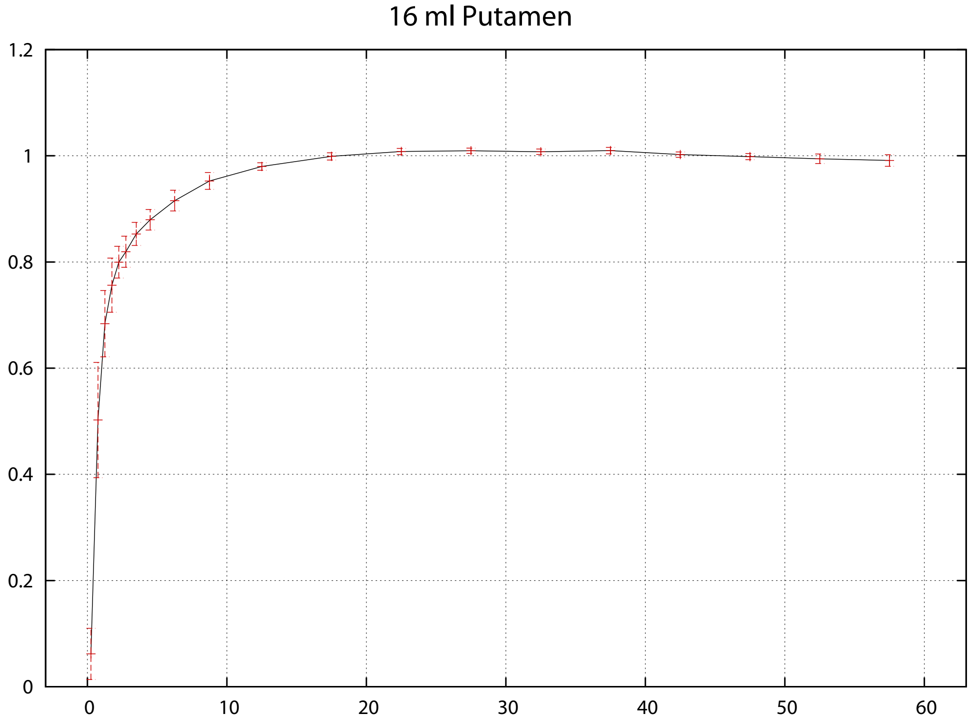
Table 14.10: *Cross comparisons of full masks for Sample 1. The blocks contain 'corr', 'dist' and 'shift' from top to bottom. Subtraction order for 'shift': left minus upper caption. Upper triangle: data based on Cortex Digests, lower: on Hippocampus Digests.*

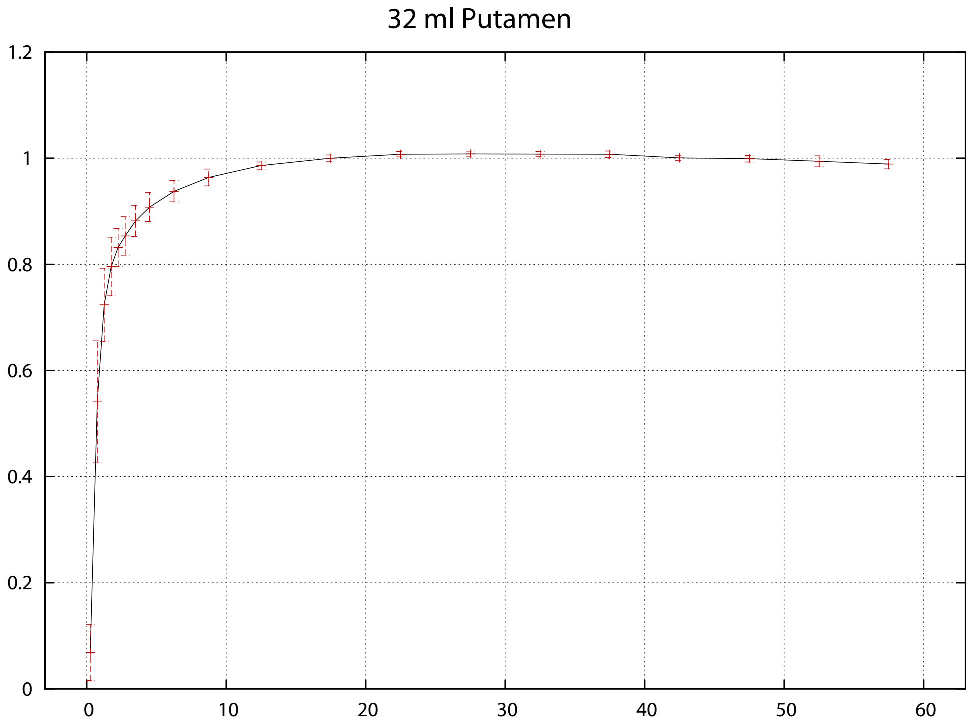
the latter are better comparable and the differences of the first 10 minutes become visible. Even so, 16 and 32 ml Putamen curves look almost identical. In Diagram 14.2 they are shown together to make it obvious that 32 ml starts with higher values and Cerebellum curves are in between.

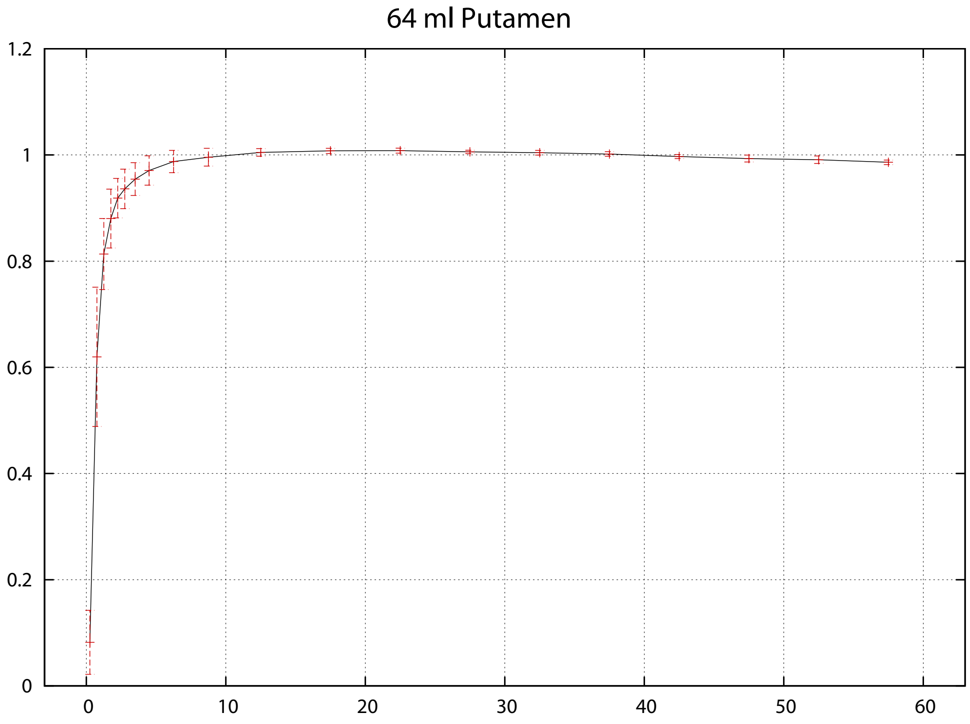
14.1.14 Washout from the Reference Region

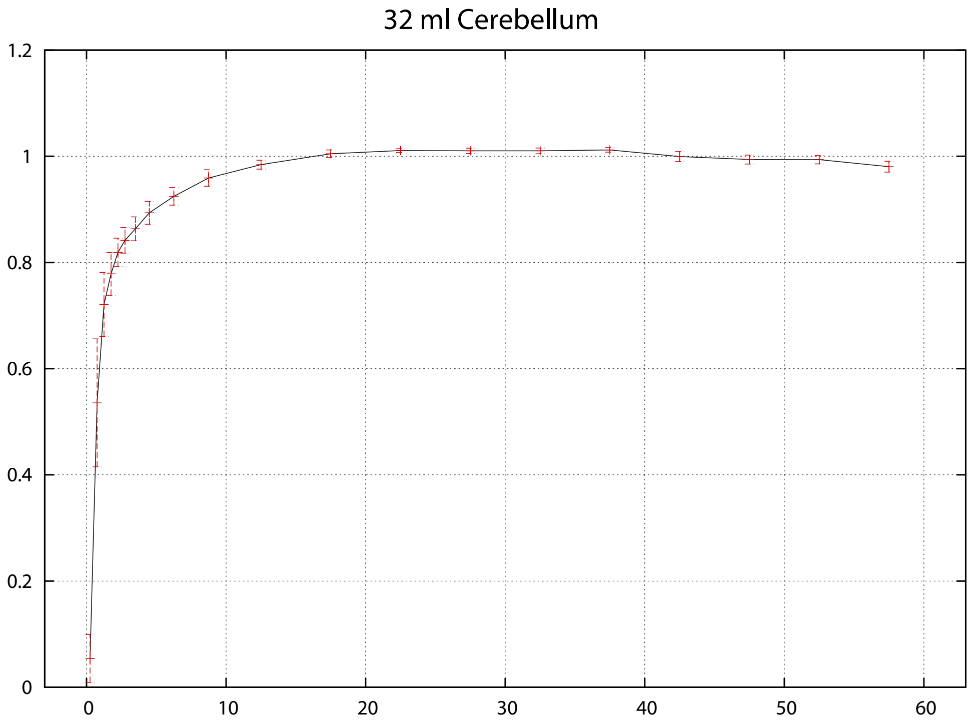
A superficial look at the plots of section 14.1.13 is enough to find that Assumption 4 ("ideal reference region", section 3.2.2) does not hold: there is obviously some loss of radioactivity from the reference region toward the end of scantime. For 16 ml Putamen we quantified the effect as follows. For each

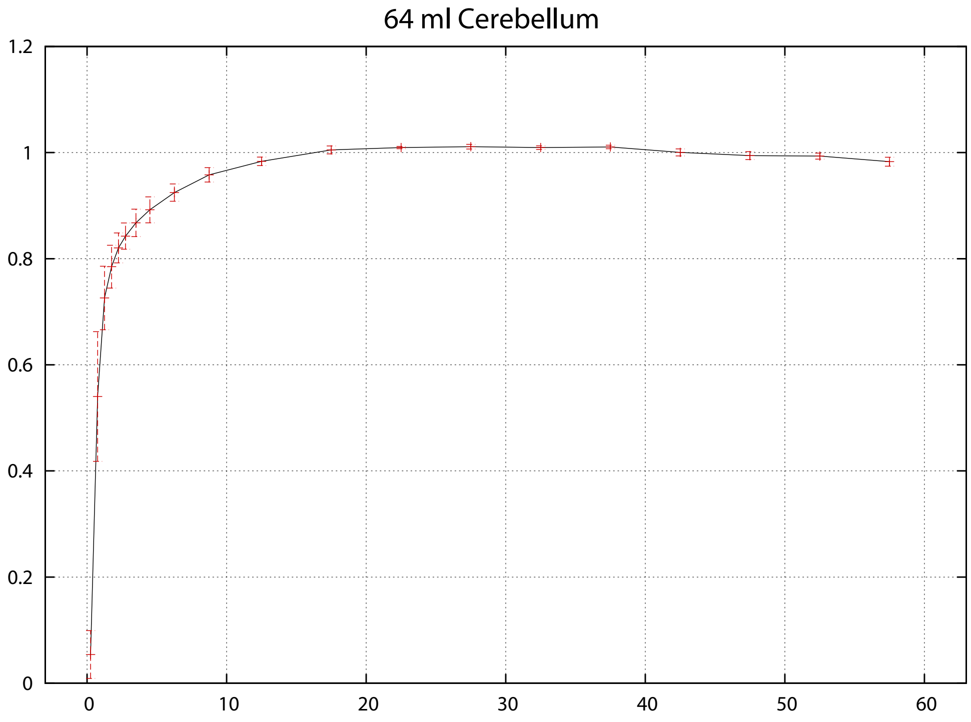

5.4 ml Putamen
1.2
1
0.8
0.6
0.4
0.2
0
0
10
20
30
40
50
60


16 ml Putamen
1.2
1
0.8
0.6
0.4
0.2
0
0
10
20
30
40
50
60


32 ml Putamen
1.2
1
0.8
0.6
0.4
0.2
0
0
10
20
30
40
50
60


64 ml Putamen
1.2
1
0.8
0.6
0.4
0.2
0
0
10
20
30
40
50
60


32 ml Cerebellum


64 ml Cerebellum

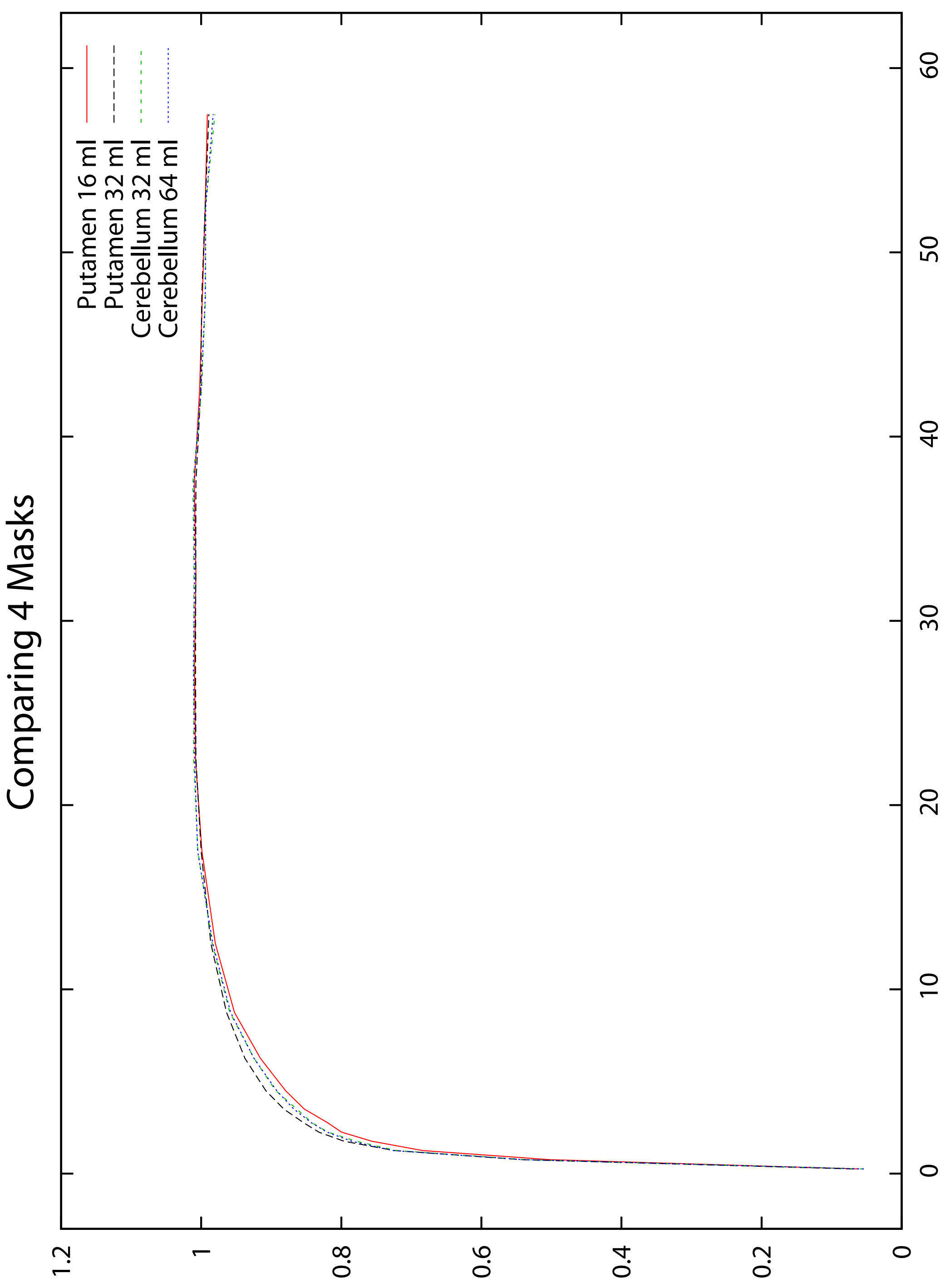


Figure 14.2: *Comparing reference curves of 4 types. The two Cerebellum curves are almost indistuingishable.*

Patients	sample size N	frames 17 to 20 in percent of frames 13 to 16	cases where frames 17 to 20 are larger	P-value of two tailed test
Sample 1	12	98.78	0	0.000
Sample 2	17	99.69	5	0.143
Samples 3 and 4	15	-	2	0.007

Table 14.11: *Studying washout of radioactivity in the reference regions*

subject, the average of frames 17 to 20 was expressed as percentage of the average of frames 13 to 16. Of the percentages, the average was computed across the patient sample, leading to figures in column 3 of **Table 14.11**. Finding them below 100 % confirms the visual impression. Column 4 reports how many patients of each sample were exceptions to this rule, and column 5 gives the significance of a two tailed binomial test of the column 4 data to prove inferiority of frames 17 to 20. In Sample 1, this is highly significant, in Sample 2 it isn't, but the union of Samples 3 and 4 (that were scanned in Milan) provides again significant evidence for washout of radioactivity from Putamen.

To exclude k_{3r}-induced deformation as a possible cause, Sample 1 and 2 reference curves were k_{3r}-corrected using k_{2r}=0.1 with k_{3r}=0.5, and the table recomputed for resulting reference curves. Percentages increased very slightly to 98.83 and 99.74 percent, and in Sample 2 there appeared one additional count in column 4. Thus, even extreme k_{3r}-correction has little effect on frames as late as 13 to 20, and does not reverse the phenomenon.

14.2 Error induced by C_t

In order to isolate the contribution of C_t to random error of k_3, we evaluated TACs from abba-a and -b images against the same reference curve, which was obtained from the corresponding abba-t image. Two parametric k_3-images were thus computed for each patient of Sample 1. Using a mask for every cerebral region, a pair of regional mean values was obtained. The pairs were processed across the sample as in section 14.1, using formula (14.3). Results were extrapolated to full amounts of radioactivity by division through $\sqrt{2}$ (see section 14.1.8). We obtained a list (column 4 of **Table 14.12**) of rough estimates of the standard error of regional mean values of k_3. A typical reading in cortex is 0.0012. Considerable variation of the estimates is seen, presumably caused by the regions' specific sizes, shapes and kinetic constants (columns 2 and 3). Obviously, regional k_3 are affected by smaller errors than their voxel based counterparts, the relation is shown in column 5. Mean values of *k3dev* (unlisted) that were used to compute these ratios, had been obtained by averaging *k3var* across all voxels of abba-a and -b images of all patients, then taking the square root.

	Voxels	k_3	k_3	=% of
Region			SD	*k3dev*
White Matter	54494	.0746	.0008	2.35
Frontal Cortex:				
Superior Frontal Gyrus	14384	.0870	.0006	3.74
Middle Frontal Gyrus	7411	.0808	.0011	7.09
Inferior Frontal Gyrus	6014	.0845	.0014	7.60
Orbito Lateral Gyrus	2237	.0795	.0013	9.78
Orbito Medial Gyrus	2828	.0879	.0016	7.07
Precentral Gyrus	6525	.0984	.0015	8.08
Gyrus Rectus	1469	.0887	.0019	9.96
total:	40868	.0871	.0006	3.20
Parietal Cortex:				
Superior Parietal Gyrus	4961	.0725	.0009	6.97
Supramarginal Gyrus	5751	.0740	.0008	6.60
Inferior Parietal Lobule	3798	.0660	.0009	8.96
Postcentral Gyrus	7315	.0886	.0012	7.28
total:	21825	.0771	.0005	4.02
Temporal Cortex:				
Superior Temporal Gyrus	8516	.0828	.0009	5.16
Middle Temporal Gyrus	6540	.0726	.0015	9.77
Inferior Temporal Gyrus	4349	.0706	.0015	9.29
Fusiform Gyrus	4428	.0901	.0013	4.48
total:	23833	.0791	.0007	3.41
Occipital Cortex:				
Superior Occipital Gyrus	2042	.0675	.0018	13.39
Middle Occipital Gyrus	2261	.0663	.0011	8.42
Inferior Occipital Gyrus	3163	.0684	.0011	7.42
Lingual Gyrus	3927	.0765	.0015	10.12
Cuneus	2378	.0650	.0009	5.94
Precuneus	3976	.0666	.0014	15.11
total:	17747	.0689	.0006	4.45
Other:				
Insular Cortex	2505	.1068	.0042	15.42
Cingulate Gyrus	6315	.0864	.0011	6.12
Hippocampus	2520	.1140	.0053	12.54
Amygdala	591	.1609	.0244	28.56
Brain Stem	1317	.1773	.0123	8.52
total cortex:	113093	.0810	.0004	2.14

Table 14.12: *k_3 random error induced by noise of C_t, and its relation to regional k3dev*

Chapter 15

Discussion

15.1 NLS in Relation to other Methods

- In 1984, Blomqvist [5] discovered a linear relation between C_t, C_r and their time integrals:

  $$C_t(t) = p_1 \cdot C_r(t) + p_2 \cdot \int\limits_0^t C_r(s)ds + p_3 \cdot \int\limits_0^t C_t(s)ds$$

  where:

  $$q_1 = p_1, \quad k_2 = -p_3 - \frac{p_2}{p_1}, \quad k_3 = \frac{p_2}{p_1}$$

  The coefficients p_1, p_2 and p_3 can therefore be determined by solving a Linear Least Squares problem. Computationally, it requires evaluating the integrals, filling a matrix and a vector with inner products and solving a 3×3 linear equation system, followed by trivial computation of q_1, k_2, k_3 from p_1, p_2, p_3. The solution obtained from noisy input is not the NLS solution (the three vectors on the right hand side span a linear space which is not $\mathcal{M}$). Drawbacks are that

  - there is no support for weighting
  - the integral of C_t must be computed for every voxel, ruling out time consuming interpolation methods.

  With trapezoidal integration, this is Nagatsuka's **RLS**-method [29], which is faster than the presented NLS implementation.

- An earlier method by the same authors [36] is called **Shape Analysis**: it relies on the fact that TACs become stationary at the end of scantime, and assumes that all tracer is hydrolyzed at this point, while at scanstart all tracer is unhydrolyzed. There is exactly one choice of k_3 that will explain the final amount of hydrolyzed tracer in accord with observed TAC. The method neither requires blood sampling nor a reference region, but it was later given up by the authors for lack of precision, in favor of RLS.

- General Basis Function Approach (**BFA**) is an attempt to reach the NLS result on a different path: model functions are tabulated *once* for a finite set of kinetic parameters. Then for every voxel, we select from this set of **basis functions** the function of best similarity (in terms of least squares) with the measurements, and determine remaining parameters by linear procedures.

  For every voxel, all basis functions must be scanned, hence their number determines the complexity of the method. So we have a tradeoff between speed (requiring that the number is not too high) and precision (requiring it not to be too low). Application of BFA to MP4A analysis would require a two-dimensional set of basis functions, using a grid of k_2/k_3 combinations. The tradeoff is then a major problem.

- **MAP** is proposed in [11]. It is a Bayesian method reducing the estimation error in high k_3 regions by investing prior knowledge of q_1 or k_2 which are assumed constant over large areas of the brain.

15.2 NLS in recent Literature

- The paper bearing most similarity to this work is [29] of 2001, where RLS is propagated as a fast alternative, and NLS is used for validation. Unlike reference based NLS as presented here, their implementation relies on arterial blood sampling and C_r is replaced with a multiexponential fit.

- In a 2008 review [18] by Ikoma, Watabe et al., NLS is considered for the reversible tracer model with arterial Input Function, saying:

  > The advantage of NLS ... is that every parameter K_1 to k_4[1] can be estimated, and that it is free from the biases and assumptions that arise in linearization and simplification. The disadvantage is that COV is large, especially in voxel-by-voxel estimation, and that the fitting procedure is computationally very expensive. Therefore, the method is useful for ROI analysis, in which parameters are estimated in a mean TAC within a ROI, but it is not practical for parametric mapping with voxel based estimation.

- Both voxel and region based weighted NLS have apparently been used in [11] in 2008, for comparison with the MAP algorithm presented by the authors, but no information is included on the NLS algorithm and its performance.

[1] K_1 is q_1 in this work, and k_4 is 0 in the irreversible tracer model

15.3 Reference Curves

Considerable impact of the reference curves on regional k_3 results was found (chapter 11), creating an extra need for validation. There is only one instance of C_r used for every parametric k_3 image, and it shifts all voxels in the same direction, i.e. toward higher or lower k_3 values, thus impacting all regional results. We used a simulation technique called k_3-digest (section 14.1.2) to quantify the up- or downshifting tendency of reference curves, and applied data splitting techniques to obtain two stochastically independent results from every patient. The data were either splitted at sinogram level to produce two equivalent image series (a,b comparisons, sections 14.1.4 and 14.1.9) or we used reference curves sampled from unilateral left and right masks (L,R-comparisons). Evaluation of such pairs of data across patient samples led to estimates of C_r-induced bias and random error for every type of mask.

15.4 Random Error induced by Noise of C_r

In order to assess this type of error, three studies were conducted on two samples of patients (section 14.1.1):

- a,b-comparisons in Sample 1
- L,R-comparisons in Sample 1
- L,R-comparisons in Sample 2

leading to the data of **Table 15.1**. It estimates regional k_3 standard error inflicted on Hippocampus (upper line in every row) and a typical cortex region (lower line). The values were computed from 'dist' entries of Tables 14.7 and 14.8, which were taken times $f/2$, where f is a factor from Table 14.4 and $\frac{1}{2} = \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}}$ where the first factor is from equation (14.3) and the second factor converts from half to full tracer dosage (section 14.1.8).

Throughout the table, C_r-induced error in Hippocampus is at least twice as high as in cortex. Both quantities are almost cut in half when the mask size is raised from 5.4 to 16 ml. In Sample 1, further improvement is seen when switching to 32 ml. Sample 2 reaches the bottom already at 16 ml and there is only 60% of reduction between 5.4 and 16 ml. Sample 1 begins with a higher L,R-reading of 0.0121 compared to 0.0076 in its a,b-study, while Sample 2 is in better agreement with the latter.

The difference between a,b- and L,R-comparisons is that the former cleanly isolates the effect of noise, while the latter is also sensitive to bias caused by patient (or scanner) asymmetries. The discrepancy in L,R- and a,b- studies of Sample 1 suggests there might be L,R-differences caused by more than just random effects.

	Putamen				Cerebellum	
	5.392 ml	12.456 ml	16 ml	32 ml	32 ml	64 ml
Sample 1	.0076	.0046	.0041	.0034	.0027	.0028
a,b-comparison	.0036	.0025	.0023	.0017	.0011	.0012
Sample 1	.0121	.0075	.0062	.0037	.0040	.0040
L,R-comparison	.0053	.0033	.0027	.0017	.0019	.0018
Sample 2	.0066	.0047	.0040	.0039	.0043	.0034
L,R-comparison	-	-	-	-	-	-

Table 15.1: *C_r-induced k_3 standard error in Hippocampus (upper lines) and average cortex (lower lines, not evaluated in Sample 2) for 3 studies and different types of masks*

We therefore combined a,b- and L,R-comparisons in such a way as to prove the presence of nonrandom L,R-differences (section 14.1.11). The statistical test returned a P-value of 0.059, just short of proving the conjecture on a 5% level of significance. The phenomenon goes away when the mask size is increased, it is therefore an attribute of the very brightest Putamen voxels (5.392 ml = 43% of the Putamen volume). Such asymmetry effects were not seen in Cerebellum. Apart from the pecularities of Sample 1, it is clear that small Putamen masks lead to noisy reference curves and should therefore be given up in favor of larger masks.

15.5 Bias induced by Noise of C_r

Since there is no absolute standard available, we chose the 674+2 Putamen mask as a reference point and considered differences with it at sample level. Entries of **Table 15.2** were computed by taking ‘shift’ values of Table 14.10 times a translation factor of Table 14.4. Even at sample level the data are largely disfigured by noise, but they consistently show negative bias in Cerebellum. Consistent behavior of Hippocampus Digests of Samples 1 and 2 is also seen for the large Putamen masks: positive bias, huge at 64 ml but already visible at 32 ml. Statistical significance could only be demonstrated

		Putamen				Cerebellum	
		12.456 ml	16 ml	32 ml	64 ml	32 ml	64 ml
Sample 1	Hippo	.0014	.0008	.0020	.0115	-.0007	-.0027
	Cortex	.0005	.0001	-.0010	-.0010	-.0007	-.0017
Sample 2	Hippo	.0013	.0019	.0038	.0145	-.0042	-.0048
	Cortex	.0004	.0006	.0005	.0020	-.0020	-.0016

Table 15.2: *Estimates of k_3 bias caused by different types of masks, against 5.392 ml Putamen as reference point*

	Putamen					Cerebellum	
	5.4ml	12.5ml	16ml	32ml	64ml	32ml	64ml
Reference: 5.4ml Putamen							
Sample mean	-	0.0010	0.0011	0.0024	0.0105	-0.0021	-0.0031
SD of sample mean	-	.00069	.00075	.00105	.00221	.00172	.00169
Sample mean / SD	-	1.5080	1.5316	2.2779	4.7412	-1.2125	-1.8019
P-value	-	0.132	0.126	**0.023**	**0.000**	0.225	0.072
Reference: 16ml Putamen							
Sample mean / SD	-1.5316	-0.3395	-	1.6536	4.6375	-2.0868	-2.5753
P-value	0.126	0.734	-	0.098	**0.000**	**0.037**	**0.010**

Table 15.3: *Statistical analysis of bias, based on Hippocampus Digests and the union of Samples 1 and 2*

by either considering Sample 2 in isolation or lumping the samples together. The latter was done for Hippocampus Digests (**Table 15.3**). Translation factors were not applied for this analysis. At patient level, we took the differences of each mask with 5.392 ml Putamen, computed their mean and sample-SD and divided the latter by $\sqrt{N}$ where N=29 is the sample size, thus estimating SD of the sample mean. The mean itself was then expressed in multiples of this quantity, and a two-tailed test, assuming Normal Distribution, applied. Significant positive bias could thus be confirmed for 32 ml and 64 ml Putamen. Negative bias of Cerebellum was not significant, but became so when moving the reference point to 16 ml Putamen (thus reducing noise). So k_3 of 32 ml and 64 ml Cerebellum masks is significantly lower than of 16 ml Putamen.

15.6 Deciding which Mask is best

5.4 ml Putamen masks as used in the past have prohibitively high levels of noise, as visible from Table 15.1. This improves when enlarging them to 12.5 or 16 ml, which is about the size of the Putamen region. Any enlargement beyond this point allows for more signal from the surroundings of Putamen, which is an undefined mix of cerebrospinal fluid and different tissue types and therefore unsuitable as reference, although we found further noise reduction in Sample 1 for the 32 ml mask.

The highest in vitro measurements of AChE, corresponding to k_{3r}=3 or 4, have been reported for Putamen, suggesting it might be better suited as reference tissue than Cerebellum with only k_{3r}=0.65 (section 13.1). In line with these reports, we found the highest intensities in reference curves sampled from small and medium Putamen masks (column 2 of Table 14.9). But high intensities may as well be caused by high k_1, so we looked at *k3dev* and the voxel yield (Table 14.9) for confirmation, and found Putamen masks of 5.4 to 16 ml ahead as well, while the 32 ml mask deteriorated and 64 ml had the worst scores in all 3 criteria. We also found 5.4 to 16 ml

Putamen masks slightly ahead of Cerebellum in all 3 disciplines, so we reach an unequivocal decision in favor of 12.5 or 16 ml Putamen masks. Having said this, we should nevertheless spend some time discussing the evidence.

k3dev as an Indicator of Bias

k3dev is computed from pairs of C_t and C_r, and indicates how well these data fit together. If they are not arising from the same Blood Input Function, *k3dev* will increase. Unlike Tables 15.1 to 15.3, *k3dev* considers information of the target region and is therefore a valuable independent source of information.

As pointed out in chapter 8, it increases in response to noise and bias both of C_t and C_r. C_t cannot explain the differences in Table 14.9 since the same set of TACs has been used for all table entries. So we are left with noise or bias of C_r. Concerning noise, it ought to be lower for larger masks. But we find the lowest *k3dev* readings for the small Putamen masks, so they must be the ones that are least affected by bias.

Considering Voxel Yield

Decrease in yield suggests NLS has difficulty finding a local minimum from its start point (1;0.1;0.1), hence the "landscape" of the target function, which is defined by C_r and C_t in combination, is abnormal. This may be due to the same set of reasons as increased *k3dev*. It is interesting to see it respond synchronously with the other markers in Table 14.9.

Considering C_r Plots of section 14.1.13

Of the parameters shown in Table 14.9, "intensity" is not visible in the plots, and, according to Theorem 4.5, has no impact on k_3. Observed differences of k_3 should therefore correspond to visible differences in the shape of C_r. This is obviously the case for the 64 ml Putamen curve. Comparing its plot with, say, 16 ml Putamen, suggests that the two relate to different Blood Input Functions, and given all other evidence, we conclude that 64 ml Putamen is the faulty version. Looking at the plots, 32 ml Putamen is slightly leaning its way, while the remaining masks, including Cerebellum, have largely congruent plots. They provide no visible explanation for the negative bias of Cerebellum compared to 16 ml Putamen. This shows how sensitive k_3 is to subtle changes in the reference curves.

15.7 k_{3r}-Correction

According to section 6.4 and Table 6.3, low k_{3r} of the reference region translates to negative bias of k_3. Hence it may well explain the negative bias seen

	k_{2r}=0.1			k_{2r}=0.3
	k_{3r}=4	k_{3r}=2	k_{3r}=1	k_{3r}=1
Average cortex	0.32	1.53	5.72	15.11
Hippocampus	0.50	2.42	8.98	23.96

Table 15.4: *Percental increase of regional mean values of k_3 in response to applying k_{3r}-correction using different combinations of k_{2r} and k_{3r} (computed from the PET scan of subject M01014)*

in comparing 64ml Cerebellum with 16ml Putamen of Table 15.2, which is somewhere between 0.0030 and 0.0060 in Hippocampus, corresponding to 2.5 to 5 percent of k_3. **Table 15.4** suggests that

- Putamen reference curves infer about 0.5 percent of negative bias if k_{3r}=4 is assumed. Note that this ignores possible contamination of C_r by Partial Volume Effects which might lead to much lower k_{3r}.
- this puts Cerebellum at anything between 3 and 6 percent of negative bias, so it would take k_{3r}-correction using k_{2r}=0.1 and k_{3r} somewhere between 1 and 2, to compensate for it.

k_{3r}-correction using k_{2r}=0.1 and k_{3r}=1.5 therefore seems appropriate with 64 ml Cerebellum masks.

15.8 Random Error induced by Noise of C_t

On the voxel level, this type of error can be estimated by *k3dev* (column 7 of Table 13.1), we found a cortical average of 0.0167. Unlike error induced by C_r, it has a chance to average out across every region. If the voxels were stochastically independent, the reduction factor should be the inverse square root of their number. But as a result of smoothing during the Gauss Filter step, they are dependent, so the reduction factor is larger than that. It depends on the size and shape of every region: the larger it is and the more distributed across the brain, the smaller the dependence between its voxels and hence, the reduction factor.

Regional C_t-induced errors were determined for **Table 14.12** using data splitting techniques. In most cortex regions they were as low as 0.0012, while 0.0053 was found in Hippocampus. Corresponding reduction factors were computed by division through regional means of *k3dev*. They ranged between 2.35 percent of the White Matter region which is large and widely distributed, and 28.5 percent in Amygdala which is small.

It is interesting to see what happens when FWHM is raised from 8 to 10 millimeters (data unlisted). Reduction factors increased, owing to more stochastic dependence. The increase fully consumed what precision was gained at the voxel level, so there was no improvement in regional results.

This corroborates our finding in section 10.6.2 that using FWHM of 10mm is inferior to using 8mm.

15.9 Bias induced by Noise of C_t

Unlike random error of the previous section, bias will be equal on the regional and the voxel level and is easiest quantified by simulations, where the true k_3 is known. It was observed that noise of C_t induces positive bias, noise of C_r negative bias. **Table 15.5** shows simulation results for recommended evaluation policies (reference volume=16ml, NLS skipping 2 frames) assuming k_2=0.1: with FWHM=8mm and for $k_3 \leq 0.2$, bias stays below 2.05 percent. Considering possible miscalibration of the simulator (section 7.8), we should raise this limit to 3 percent. With FWHM = 6 mm, the bias roughly doubles, with 10 mm, it is cut in half. If k_2=0.05 is assumed instead of 0.1, it triples (data unlisted). In real PET data, we have a mix of C_t-induced bias which is positive with C_r-induced and Discretization Bias which are negative. The three types are additive and therefore partly cancel out. The table shows that discretization- and C_r-induced bias are less dependent on k_3 than C_t-induced bias. The same applies to k_2 (data unlisted).

	k_3		
	0.08	0.12	0.2
C_t-induced Bias [%]:			
FWHM=6mm	0.77	1.57	3.66
FWHM=8mm	0.31	0.65	2.05
FWHM=10mm	0.16	0.32	1.05
Discretization Bias [%]:			
	-1.38	-1.45	-1.54
C_r-induced Bias [%]:			
V=5.392 ml	-0.60	-0.77	-1.09
V=16 ml	-0.20	-0.25	-0.36
V=64 ml	-0.04	-0.06	-0.08

Table 15.5: *Bias of 3 different types, quantified by Double Random Simulations. For simulating C_t-induced bias in isolation, the reference volume V was set to 128 ml. For simulating C_r-induced bias, FWHM was set to 50 mm. 33,333 iterations per entry were performed. Discretization Bias was simulated from noiseless TACs as in section 6.5.1, and subtracted from the other simulation results in order to isolate every type of bias. It could be empirically shown (data unlisted) that C_t- and C_r-induced bias are roughly additive.*

15.10 Limitations of the Model

We found significant evidence for a slight washout of radioactivity from Putamen toward the end on scantime (section 14.1.14), contradicting Assumption 4 (“ideal reference region”, section 3.2.2). The same has been reported by A. Varrone [37] for 120 minute scans of monkeys. The standard model of Figure 1.2 is therefore inadequate, even in its refined version: we found a new type of systematic bias that can *not* be undone by k_{3r}-correction (section 14.1.14) and is therefore not k_{3r}-induced.

Another occasion where we encountered non-model-compliant TACs has been discussed in section 12.1ff., it is caused by blood volume effects.

15.10.1 Refining the Model?

Whenever a source of bias is identified, there is the option of including it in the model, and fit the refined model to the data. We discuss the implications of this idea in each of the three cases:

- **Nuclide washout.** The idea of irreversible nuclide accumulation in Putamen is not realistic to begin with. Every carbon atom that goes in will eventually find its way out, after sufficient metabolization, unless Putamen was to act as a final deposit, which is not very likely. All we can hope for is that washout be delayed until after the end of scantime.

  But as we have seen this is not the case. To model the washout, we could use an existing reversible tracer model, adding k_4 and k_{4r} arrows to Figure 1.2. But what if the product of hydrolysis can diffuse back to the plasma without further metabolization? Or else, there might be a mix of such back diffusion with a number of different biochemical pathways? We are looking at an effect whose mechanism is not understood, hence we do not know how to model it.

- **Blood Volume Effects.** Considering them would require including a blood term into model equations (3.1). The signal is no longer proportional to C_t, but to $(1-\alpha) \cdot C_t + \alpha \cdot (C_{pl} + C_{ph})$, where α is a factor denoting the local “blood volume” (i.e. percentage of tissue that is filled with blood), and C_{ph} is the concentration of hydrolyzed tracer in blood. The latter has hitherto not been considered. Ignoring it is not advisable since tracer hydrolysis in blood happens fast, so a considerable percentage of the blood signal comes from hydrolyzed tracer. The model would thus require a second input function which is not available from the PET data. It could be obtained by investing a fixed blood hydrolysis curve compiled from arterial blood sampling data. α would induce a fourth degree of freedom to the model. To make matters worse, much of the blood volume is venous blood whose authentic tracer concentration is not C_{pl}, but also depends on tracer extraction minus backflow during passage through tissue, where the backflow depends on how much tracer the tissue contains. Distinguishing arterial and venous blood would lead to five degrees of freedom. With every degree of freedom added, the model becomes more sensitive to noise. It therefore appears more prudent to embrace a skip2 policy than to refine the model.

- k_{3r}**-induced bias.** Its compensation is the only refinement that was undertaken (section 15.7). It is easy to apply (section 3.2.3) and does not introduce new degrees of freedom. But it requires knowledge of k_{2r} and k_{3r} of the reference region, which can only be estimated. The further they are away from the "ideal" values 0 and ∞, the larger is the error resulting from miscorrection (Table 6.4), we are therefore well advised to use k_{3r}-correction as little as possible.

Chapter 16

Conclusions

16.1 VINCI Implementations

- The Herholz-Zuendorf method of MP4A analysis ("COLOGNE method") has been re-implemented in C++ as a VINCI-plugin, automatizing a complex workflow involving image normalization, coregistrations, image preprocessing and kinetic analysis that used to require VINCI, SPM99/MATLAB and human intervention.
- As a second implementation, we provided the "maskless MP4A tool", containing an independent algorithm for kinetic analysis and various other policy changes by the author, see sections 16.2 through 16.6.

16.2 Voxel based NLS

We made weighted Nonliner Least Squares fitting (NLS) available as a method for voxel based kinetic analysis (chapters 4 and 5). Its implementation as part of VINCI 3.x is powered by a built-in Gauss Newton solver. Correctness of the procedure has been established empirically using simulated data with varying levels of noise. It turned out that a single starting point at (1;0.1;0.1) could be used in all cases. The number of iterations can be kept below 20 owing to fast convergence of the method and low dimensionality of the problem (section 5.3, Table 12.2).

Filling the Jacobi matrix is the time-critical step. We took advantage of spline interpolation of the reference curve. Hence $\mathcal{F}$ and its partial derivatives are integrals of piecewise 4^{th} order exponential polynomials that can be solved analytically. This could be organized in such a way as to have linear complexity in the number of frames. We reached computation speeds of up to 5000 voxels per second on a 2.67 GHz single processor machine.

16.3 Voxel based Error Estimates

We developed an estimate *k3var*[1] of the variance of k_3, based on linearizing the model in the vicinity of the NLS solution and considering its distance from the measurements. It can be computed rapidly from pairs of C_t and C_r, taking advantage of the partial derivatives that are available as by-products of the Gauss Newton procedure.

k3var has been shown to predict the variance of simulated noisy data correctly for a wide range of simulator settings. So it provides a vehicle transferring error estimation from the simulator environment to real PET images. It has been used in this function in other parts of this work, to assist in optimizing image preprocessing (chapter 10) and reference masks (sections 14.1.12 and 15.6). It may also lend a hand in regional error estimation (section 16.7.2).

16.4 k_{3r}-Correction

is a novel technique for reconstructing ideal from measured reference curves (section 3.2.3). It requires that k_{2r} and k_{3r} of the reference region be known. Upon validation with noiseless synthetic data (section 6.4) it reduced bias by one to two orders of magnitude.

An implementation for real image data has been provided as an optional feature of the maskless MP4A tool. We do not recommend it with Putamen masks whose k_{3r}-induced bias is expected below 1 percent, but it should be used with the 64 ml Cerebellum mask in order to compensate for its negative bias (section 15.7).

16.5 Application Range

We discontinued the previous policy of voxel preselection using the Zuendorf Mask. It restricted kinetic analysis to 45.9% of the brain, excluding such regions of interest as Hippocampus and cutting through cortex areas as well. Evaluation is now attempted for every voxel of the brain, resulting in 85% to 95% of yield, depending on the patient. More than 99% are reached in most cortex regions, and between 94% and 99% in Hippocampus.

16.5.1 Image Preprocessing

The previous policy of Gauss filtering while protecting low intensity regions from hot neighborhoods has been preserved. The Zuendorf Mask used to play a crucial role in these procedures, this has now been taken over by a sequence of 10 to 15 adaptive masks obtained by intensity thresholding

[1] and its square root, *k3dev*

of the PET data (section 10.4.3). The inner border of the Zuendorf Mask used to be a source of artefacts. This problem has been addressed by keeping voxels at a distance from the border of their respective filter masks whereever possible (section 10.7).

16.5.2 Reference Mask Generation

The previous policy of using a fixed Putamen mask of 674 voxels has been replaced by an adaptive strategy of mask generation from a precursor mask, based on erosion or dilation and intensity thresholding (section 10.3). Masks can be generated for Putamen or Cerebellum at any given size, and they will compensate for differences in patient anatomy or coregistration errors.

16.6 Optimizing Procedures with Respect to Noise and Bias

- Framing: the “traditional” schedule of 20 frames (section 6.1.1) has proved to be a favorable choice. In simulations, its stability with respect to noise was found to be equivalent or superior to every other 20-frame schedule. Minor improvements of stability could only be made by raising the number of frames substantially. But this was not tried with real images where it seems counter-indicated since it might compromise image reconstruction by diluting the signal too much.

- Placement of time points: the former policy of mapping them to the frame ends has been given up in favor of the frame centers. In simulations, bias could thus be reduced from 20% to 1%.

- Frame Weighting: we compared different weighting strategies in simulations regarding stability with respect to noise. For the COLOGNE method, C_r- proportional weighting that had been used by the Zuendorf implementation could be improved by combining it with Decay Weighting (section 2.7). The combined strategy was clearly superior in simulations. COLOGNE thus performed on the same level as the competing NLS method. For the latter, we recommend Decay Weighting for the sake of simplicity. Equivalent performance was reached with the combined strategy.

- Gauss Filter width 'FWHM': 10mm that were used in the past could be replaced by 8mm without compromising quality of regional results. This improves resolution of k_3 parametric images and reduces bias induced by Partial Volume Effects.

- Skipping frames: we recommend to exclude the frames of the first minute from kinetic analysis in order to avoid bias induced by blood volume effects. It could be demonstrated by comparison of COLOGNE and NLS that this policy considerably stabilizes results.

- Reference mask volume: we found that 5.4 ml masks as used in the past allow for too much noise in the reference curves (see the following section). We showed that volumes of 12.5 to 16 ml are a much better choice, and Cerebellum masks of 32 or 64 ml combined with appropriate k_{3r}-correction are a viable alternative.

16.7 Regional Error Estimates

Quantification of regional error is nontrivial since the Gauss Filter step induces stochastic dependence between voxels. So we have no translation formula between voxel based and region based error. Instead, investigations had to be based on patient samples and data splitting techniques at the sinogram level (chapter 14). For C_r-induced errors, we also applied left/right comparisons and a hybrid technique combining simulations with measured data (section 14.1.2).

16.7.1 Error induced by C_r

We discovered a major source of error that had previously been ignored: noise of the reference curves. We were alerted to the problem by finding large differences in regional k_3 results depending if Putamen or Cerebellum had been used as reference (chapter 11). This came as an uneasy late surprise during the final stage of this work, so the author was given extra time to eliminate as much of this problem as possible and provide an assessment for the rest.

Random Error

Using data splitting techniques and left/right comparisons, it could be shown that 5.4 ml Putamen masks are to blame for most of the problem, inducing k_3 standard errors of up to 0.0121 in regional results of Hippocampus and up to 0.0053 in typical cortex regions (Table 15.1). Corresponding errors obtained with 32 ml Cerebellum masks are 0.0040 and 0.0019. Improvement could also be reached by enlarging Putamen masks to about the size of Putamen (16 ml), leading to standard errors of about 0.0050 and 0.0025[2]. Even so, C_r-induced error remains the major single source of uncertainty.

While the above is based on image data of two patient samples, we found random errors of similar size in C_r Block Simulations (data unlisted).

[2] obtained by averaging a,b- and L,R-results of Table 15.1

Bias

The contribution of noise of C_r to bias of k_3, as measured in simulations, is low (see section 16.7.2). Larger systematic deviations were observed when comparing C_r from PET image data sampled with different masks. While quantification is difficult, it could be shown based on a sample of 29 patients that, compared to 16 ml Putamen masks, 32ml and 64ml Cerebellum masks lead to negative bias (see Table 15.3). The difference is roughly 0.0030 to 0.0060 k_3-units in Hippocampus and 0.0020 in cortex (by comparing columns of Table 15.2). The bias can be undone by appropriate k_{3r}-correction (sections 16.4 and 15.7).

Placing a zero point on the bias scale remains voluntary since the true in-vivo k_3 values are unknown. But we found indications based on *k3dev* (see section 15.6) that 16 ml Putamen gives a slightly better representation of the patient's Blood Input Function than Cerebellum with or without k_{3r}-correction.

16.7.2 Error induced by Noise of C_t

Random Error

This is different from C_r-induced error in that it averages out across a region. Depending on the patient, voxel based error levels range between 0.0100 and 0.0210 in average cortex, 0.0290 and 0.0620 in Hippocampus[3]. For every region of the brain, there is a reduction factor indicating how voxel based errors translate to regional errors. It depends on size and shape of the region and FWHM used during image preprocessing. Rough estimates of these factors were obtained from a sample of 12 patients using data splitting techniques (column 5 of Table 14.12). They might be useful for obtaining regional error estimates for individual patients when multiplied with their regional *k3dev* mean values, but it would take more data and work to corroborate this claim.

The C_t-induced contribution to regional k_3 random error was 0.0012 on average in cortex regions and 0.0053 in Hippocampus (Table 14.12). In cortex, this is clearly below the C_r-induced contribution, in Hippocampus it is comparable. Error levels in "hotter" areas are much higher. Regional errors could *not* be reduced by raising FWHM from 8 to 10 mm.

Bias

was isolated in Double Random Simulations and found to be always positive (section 15.9). It depends on k_2 and k_3 and stays below 3 percent for combinations of practical interest (section 15.9). In real situations, it

[3] data from Sample 1

partly cancels out with C_r-induced and Discretization Bias which are negative and of similar size. Total bias may therefore come out with either sign, depending on k_2, k_3, FWHM and the reference volume.

16.8 Resolving an Inconsistency of COLOGNE

COLOGNE showed conflicting results in parts of the brain depending if 2 leading frames were excluded from evaluation (skip2 policy) or not. We implemented NLS in order to decide which result was correct. As expected, NLS showed better stability, and it led to a clear decision in favor of the skip2 result. It could be shown that the problem was caused by blood volume effects, i.e. signal emanating directly from the blood during the first tracer passage through the brain (section 12.1 ff.). Since the model does not account for signal from the blood, we are faced with non-model compliant TACs and ensuing bias - unlike any other type of bias discussed above. Refining the model is impractical (section 15.10.1), the skip2 policy appears more sensible. It is interesting that this policy was already followed by G. Zuendorf, and that his brain mask excludes the problematic regions.

Appendix A

Supplements

A.1 Optimal Summation

Theorem A.1 *Given N independent random variables X_1,...,X_N with positive expectancies a_1,...,a_N and having their variances proportional to these expectancies, we seek nonnegative factors m_1,...,m_N that will minimize the relative error of the linear combination $S := \sum_{i=1}^{N} m_i \cdot X_i$. Contention: the m_i must equal the same constant, whose value does not matter.*

Proof:
Let k be the constant of proportionality, so the variances are ka_i,...,ka_N. Then $m_i \cdot X_i$ is of variance $m_i^2 ka_i$ for i=1,...,N, and because of independence, $\sigma^2(S) = \sum_{i=1}^{N} m_i^2 ka_i$ is the variance of S. So the quantity we wish to minimize is given by the function

$$F(m_1, \ldots, m_N) = \frac{\sqrt{\sum_{i=1}^{N} m_i^2 ka_i}}{\sum_{i=1}^{N} m_i a_i} = \sqrt{k} \cdot \frac{\sqrt{\sum_{i=1}^{N} m_i^2 a_i}}{\sum_{i=1}^{N} m_i a_i}$$

Its partial derivative with respect to m_j is

$$\frac{\partial F}{\partial m_j} = \sqrt{k} \left(\frac{\frac{m_j a_j \sum m_i a_i}{\sqrt{\sum m_i^2 a_i}} - a_j \sqrt{\sum m_i^2 a_i}}{\left(\sum m_i a_i\right)^2} \right)$$

We want the global minimum on the domain $m_i \geq 0, i = 1, \ldots, N$ with at least one m_i greater than 0 (otherwise F is undefined). There is no local minimum anywhere on the borders: letting m_j=0 in the above formula results in a negative value, so every border point has smaller values in its vicinity. So we check the interior. Equating the partial derivative with zero brings

$$a_j\sqrt{\sum m_i^2 a_i} = \frac{m_j a_j \sum m_i a_i}{\sqrt{\sum m_i^2 a_i}}$$

$$\sum m_i^2 a_i = m_j \sum m_i a_i$$

$$\sum_{i=1}^{N}(m_j - m_i)m_i a_i = 0$$

In a local minimum, this will hold for j=1,. . . ,N. Let us assume the equation holds at some fixed point and let m_j be its maximum component, making the brace nonnegative in all summands. Since both the m_i and a_i are positive, the expression can be zero only if all braces vanish, which implies $m_j = m_i$ for all i, hence the m_i are equal.

By inspection it is clear that F is a homogenous function, meaning that its value doesn't change if all arguments are scaled by the same factor. Hence the value of the m_i has no bearing on the outcome.
Existence:
due to the homogeneity of F, every value taken on our domain is also taken at some point of its intersection with the sphere $\sum m_i^2 = 1$, which is a compact set. Since F is continuous, it takes a global minimum on that set. Because of homogeneity, that minimum is also global on our domain.

A.2 Investigations on Convergence

On subject M01014, k_3 images were computed using different break-off thresholds l_1=l_2 ranging from 10^{-4} to 10^{-8} in a geometric sequence. Results are in **Table A.1**. Column 2 has the number of failing voxels. Up to 10^{-7}, there is little change, followed by rapid increase towards 10^{-8} which is probably due to the limitations of single precision float arithmetics[1]. The average number of iterations of the (non failing) voxels of cortex and Hippocampus was taken. It rises steadily in response to sharpening the break-off criterion. The increase (columns 5 and 7) can be looked upon as additional iterations needed to reduce step sizes in the late convergence phase. It appears almost constant, indicating linear convergence. For an order of magnitude, r=1.16 and r=1.46 iterations[2] are needed for cortex and Hippocampus, respectively. One iteration then corresponds to an error reduction factor of $\frac{1}{10^{1/r}}$ which is $\frac{1}{7.28}$ in cortex and $\frac{1}{4.84}$ in Hippocampus. There were also voxels of much slower convergence, requiring up to 200 iterations as shown in **Table A.2**. Using a limit of n_1=200 iterations and l_1=l_2=10^{-7} as break-off thresholds, the voxels were classified by iteration number (i.e. number of iterations needed to meet the break-off criterion). Column 2 has the percentage of voxels in each class, cumulated percentages are in column 3. They show that there are 7.55 percent of voxels requiring more than 20 iterations. Column 4 ($noise_{rel}$ of section 4.7.2) shows that noisiness of TACs increases

Break-off threshold	voxels failing to converge		Iterations, averaged over region:			
			cortex		Hippocampus	
l_1=l_2 =	absolute	% of brain	*iter*	increment	*iter*	increment
$\sqrt{10} \cdot 10^{-4}$	19677	7.82	4.16		4.93	
10^{-4}	19788	7.87	4.71	0.55	5.64	0.71
$\sqrt{10} \cdot 10^{-5}$	19864	7.90	5.29	0.58	6.37	0.73
10^{-5}	19931	7.93	5.87	0.58	7.08	0.71
$\sqrt{10} \cdot 10^{-6}$	19998	7.95	6.45	0.58	7.84	0.76
10^{-6}	20066	7.98	7.03	0.58	8.57	0.73
$\sqrt{10} \cdot 10^{-7}$	20238	8.05	7.63	0.60	9.32	0.75
10^{-7}	20955	8.33	8.21	0.58	10.07	0.75
$\sqrt{10} \cdot 10^{-8}$	39988	15.90	8.94	0.73	10.94	0.87
10^{-8}	175230	69.68	9.76	0.82	11.59	0.65

Table A.1: *Failrate and convergence speed in response to break-off threshold. From subject M01014, using skip2 policy and an upper limit of 200 iterations*

[1] step sizes were calculated using single precision floats, while the iterations themselves were performed in double precision arithmetic

[2] 2 times 0.58 or 0.73 as seen in the table

with iteration number. These are voxels of high k_2 and k_3, as is visible from columns 5 and 6, leading to high error levels (column 7). *simcr*, q_1 and the overall voxel intensity (section 10.2) were checked for similar correlations (data unlisted), but none were found.

Iteration number	percentage of converging voxels	(cumulated)	Class averages of:			
			$noise_{rel}$	k_2	k_3	*k3dev*
3	0.04	0.04	0.0465	0.1074	0.0924	0.0103
4	1.77	1.81	0.0511	0.1020	0.0870	0.0107
5	8.72	10.53	0.0533	0.0965	0.0828	0.0111
6	15.13	25.66	0.0566	0.0932	0.0807	0.0122
7	16.08	41.74	0.0603	0.0922	0.0815	0.0141
8	13.26	55.00	0.0649	0.0929	0.0861	0.0182
9	9.42	64.42	0.0705	0.0965	0.0965	0.0262
10	6.90	71.32	0.0739	0.1013	0.1103	0.0371
11	5.09	76.41	0.0780	0.1156	0.1242	0.0443
12	3.84	80.25	0.0811	0.1231	0.1375	0.0594
13	2.88	83.13	0.0838	0.1347	0.1539	0.0684
14	2.25	85.38	0.0887	0.1655	0.1696	0.0789
15	1.79	87.17	0.0901	0.1712	0.1805	0.0867
16	1.45	88.62	0.0950	0.2006	0.1951	0.0996
17	1.19	89.81	0.0946	0.2340	0.2154	0.1116
18	1.02	90.83	0.0953	0.2498	0.2311	0.1398
19	0.87	91.70	0.0977	0.2822	0.2418	0.1346
20	0.75	92.45	0.0955	0.2908	0.2639	0.1483
21-30	4.06	96.51	0.1013	0.3603	0.2869	0.1680
31-50	2.29	98.80	0.1073	0.4803	0.3371	0.2152
51-200	1.19	99.99	0.1088	0.4547	0.3381	0.2304

Table A.2: *Voxel properties by iteration classes. From PET data of subject M01014, skip0 policy. 19387 voxels (7.7% of the brain) failed to converge.*

A.2.1 Choosing n_1, l_1, l_2

The objective is to speed up the procedure while maintaining voxel yield and precision at reasonable levels. n_1=20 was chosen with speedup in mind. According to Table A.2, it results in a voxel loss of 7.55 percent. By introducing l_2=10^{-4} to facilitate break-off after n_1=20 iterations, that rate could be reduced to 1.82 percent. It remains to investigate if precision is not compromised too much by this measure. Of the 5.73 percent voxels regained, we compared their k_3 values under the new and the previous regime (which had n_1=200, l_2=l_1=10^{-7}), and found maximum differences of 0.00038, which was considered tolerable.

A.3 On the Benefits of Step Size Control

The unmodified Gauss Newton method was found to fail unnecessarily on certain sets of synthetic data. In particular, when running on TACs simulated for k_1=k_{1r}=1, k_2=0.04 and k_3=0.01, NLS had a failrate of some 90%, while COLOGNE returned good results. The phenomenon became even more pronounced as the amount of simulated noise was reduced. It turned out that the iteration path first proceeds through negative territory (with at least one constant below 0), then bounces back and forth and misses the attractor.

The problem can of course be fixed by providing S_5=(1;0.04;0.01) as a supplementary starting point, to be used whenever a run from S_1 has failed. One could thus install a whole sequence of starting points if needed. Every one of them would act as an additional chance for accepting a voxel, at a price paid in terms of computation time.

Since negative intermediates have been found at the heart of the problem, prohibiting them is another obvious idea. It led to the lambda strategy presented in section 5.4, keeping the iteration path well inside the octand where q_1, k_2, k_3 are positive. But unlike supplementary starting points which are guaranteed to raise the evaluation yield, there is no such guarantee for the lambda strategy. It is entirely possible that the octand borders become competing attractors, inviting failure or bad results in cases where unmodified Gauss Newton had been successful.

For the purpose of validation, we enabled our implementation to process multiple parameter sets, each containing λ, n_1, l_1, l_2 and a starting point. If a run with one strategy failed, it would load the following parameter set and try again. This made it possible to compare the λ-strategies with those based on additional starting points. It turned out that setting λ to 0.9 obliterated the need for the latter. More exactly: few multiple runs would fail with a λ=0.9 strategy from S_1=(1;0.1;0.1), and then go on to succeed with a different starting point. Also, no cases were seen where λ=0.9 failed and then λ=0 succeeded from the same starting point. Thus, λ=0.9 was found to increase evaluation yield just like additional starting points, but at no extra cost.

A.4 Privileged Area

The regions included in the “Privileged Area” of section 10.5.1, are defined by the atlas of section 10.1. **Table A.3** shows their names and sizes (in percent of the full brain mask) and names:

Name	Volume fraction of brain
White Matter	21.67 %
Paraventricular Nucleus	0.38 %
Supramarginal Gyrus	2.29 %
Superior Frontal Gyrus	5.72 %
Precentral Gyrus	2.59 %
Cingulate Gyrus	2.51 %
Middle Frontal Gyrus	2.95 %
Superior Temporal	3.39 %
Inferior Frontal Gyrus	2.39 %
Inferior Occipital Gyrus	1.26 %
Cuneus	0.95 %
Lateral Fronto-Orbital Gyrus	0.89 %
Insular Cortex	1.00 %
Superior Parietal Gyrus	1.97 %
Pre-Cuneus	1.58 %
Middle Temporal Gyrus	2.60 %
Lingual Gyrus	1.56 %
Postcentral	2.91 %
Middle Fronto-Orbital Gyrus	1.12 %
Gyrus Rectus	0.58 %
Hippocampus	1.00 %
Superior Occipital Gyrus	0.81 %
Fusiform Gyrus	1.76 %
Entorhinal Area	0.16 %
Middle Occipital Gyrus	0.90 %
Inferior Parietal Lobule	1.51 %
Inferior Temporal	1.73 %
other	0.19 %

Table A.3: Composition of “Privileged Area”

A.5 Decay Fractions

The results of section 7.1.3.3 were computed from columns 2 to 5 of **Table A.4**:

	Brain	Decay Fractions:		
Subject	Volume	whole	Putamen	Cerebellum
Code	[ml]	Brain	(1ml)	(1ml)
M01001	1394.25	0.03560	0.00599	0.00597
M01002	1470.89	0.03887	0.00693	0.00549
M01003	1682.11	0.03798	0.00564	0.00517
M01004	1737.61	0.04301	0.00701	0.00635
M01005	1553.1	0.01689	0.00289	0.00256
M01007	1482.63	0.06692	0.01210	0.01175
M01008	1214.54	0.04701	0.00963	0.00789
M01009	1473.75	0.02324	0.00390	0.00354
M01010	1498.1	0.01697	0.00269	0.00234
M01012	1347.5	0.05117	0.00957	0.00794
M01013	1313.2	0.03630	0.00757	0.00609
M01014	1392.7	0.04211	0.00676	0.00658

Table A.4: *Individual Decay Fractions of Sample 1*

From Samples 3 and 4 that were scanned in Milan, the following results were computed:
Brain Volume: 1389 ± 121 ml
Putamen Decay Fraction: 0.0000728 ± 0.0000204
Cerebellum Decay Fraction: 0.0000680 ± 0.0000164
Whole Brain Decay Fraction: 0.03928 ± 0.00708

A.6 Sample 2 Cross Comparisons

Of Sample 2, Cortex Digests were not computed. **Table A.5** contains cross comparisons of Hippocampus Digests.

	Putamen				Cerebellum	
	5.392	16	32	64	32	64
	ml	ml	ml	ml	ml	ml
Putamen	-	.9724	.9462	.7879	.7549	.7170
5.392 ml	-	.0031	.0047	.0140	.0086	.0097
	-	-.0015	-.0029	-.0110	.0032	.0036
Putamen	.9724	-	.9740	.8029	.7650	.7123
16 ml	.0031	-	.0031	.0127	.0092	.0105
	.0015	-	-.0014	-.0095	.0046	.0051
Putamen	.9462	.9740	-	.8944	.7764	.7397
32 ml	.0047	.0031	-	.0104	.0095	.0107
	.0029	.0014	-	-.0081	.0060	.0065
Putamen	.7879	.8029	.8944	-	.7060	.6907
64 ml	.0140	.0127	.0104	-	.0173	.0180
	.0110	.0095	.0081	-	.0141	.0146
Cerebellum	.7549	.7650	.7764	.7060	-	.9877
32 ml	.0086	.0092	.0095	.0173	-	.0022
	-.0032	-.0046	-.0060	-.0141	-	.0005
Cerebellum	.7170	.7123	.7397	.6907	.9877	-
64 ml	.0097	.0105	.0107	.0180	.0022	-
	-.0036	-.0051	-.0065	-.0146	-.0005	-

Table A.5: *Cross comparisons of Hippocampus Digests of full masks of Sample 2. The blocks contain 'corr', 'dist' and 'shift' from top to bottom. Subtraction order for 'shift': left minus upper caption.*

Appendix B

Derivation of *k3var*

B.1 The Idea

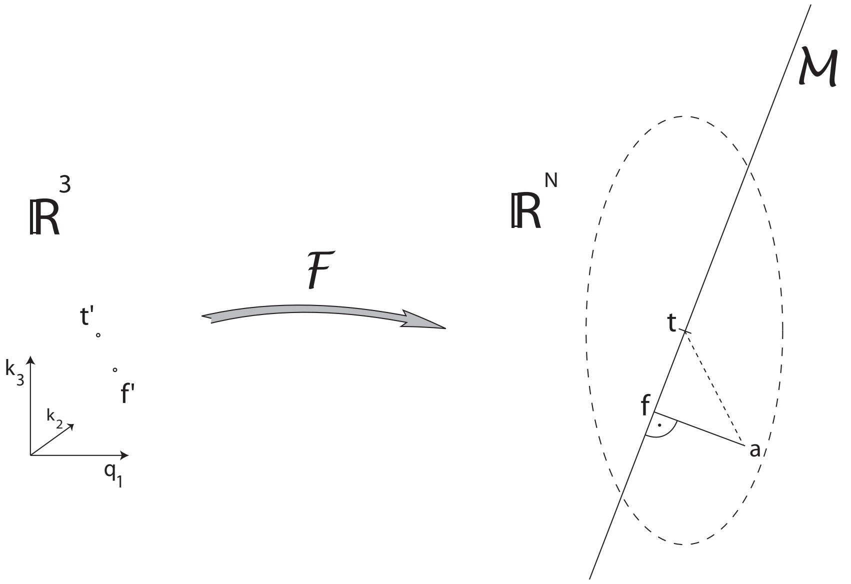


Figure B.1: *Propagation of noise to error of* k_3

Consider the set $\mathcal{M}$ of Model Compliant curves (section 4.2 and Figure 4.2), embedded in $\mathbb{R}^N$ where N is the number of frames. A magnified view of the situation is Figure B.1. Consider the triple **t'** of "true" kinetic constants, and its Model Curve **t**=$\mathcal{F}$(**t'**). The estimate is based on linearization in **t**, replacing $\mathcal{M}$ with its own tangential space and $\mathcal{F}$ with a linear parametrization of that space.

The measured, noisy, TAC is **a**. **x**=$\vec{\mathbf{ta}}$ is the noise vector. **f** is the foot of a perpendicular dropped from **a** onto $\mathcal{M}$. Its preimage **f'** under $\mathcal{F}$ is the NLS result: f'=(q_1,k_2,k_3). It is different from the true set of constants **t'**, the difference reflecting the mismatch between **f** and **t**. Within $\mathcal{M}$, there is one vector pointing in direction of the steepest ascent of k_3. It will be called the $\mathbf{k_3}$**-gradient**. For its length, we assign the speed at which k_3 changes when moving in that direction.

The projection of $\vec{\mathbf{ta}}$ on the $\mathbf{k_3}$-gradient accounts for the error of k_3. Its projection orthogonal to $\mathcal{M}$, which is $\vec{\mathbf{fa}}$, can be measured, since f (or rather its preimage f') is found by NLS. If we knew the average relation between the two projections, we could compute the first from the second and thus obtain the desired result. That relation can be obtained by modelling the noise cloud, i.e. the distribution of $\vec{\mathbf{ta}}$.

Its distribution density is normal in each coordinate. If its standard deviation were the same in all coordinates (**isotropic noise**), this would make a noise cloud of perfect spherical symmetry as explained in section 2.7.4. So we start with this special case and generalize it in the following section.

Owing to spherical symmetry, the squared projections of $\vec{\mathbf{ta}}$ distribute evenly onto the vectors of any given orthonormal base of $\mathbb{R}^N$. Consider an orthonormal base containing the normalized k_3-gradient, and N-3 vectors spanning the orthocomplement of $\mathcal{M}$. Then the k_3-gradient gets 1/N of the squared projections, and the orthocomplement gets (N-3)/N. So the squared projection on the k_3-gradient is, on average, 1/(N-3) of $\|\vec{\mathbf{fa}}\|^2$:

$$p^2 = \frac{\|\vec{\mathbf{fa}}\|^2}{N-3} \tag{B.1}$$

where p is the length of the projection. Let Δk_3 be the computed minus he true k_3, we have $\Delta k_3 = p \cdot \|k_3\ gradient\|$. Taking both sides of (B.1) times $\|k_3\ gradient\|^2$ gives

$$k3var_i = (\Delta k_3)^2 = \frac{\|k_3\ gradient\|^2 \cdot \|\vec{\mathbf{fa}}\|^2}{N-3} \tag{B.2}$$

The average of a large number of $(\Delta k_3)^2$ happens to be the variance of the k_3[1], so $k3var_i$ is an estimator of the variance.

B.2 The Details

Next we specify how the k_3-gradient is computed, and what can be done to model the noise cloud in a more realistic way. For both ends, we need to construct an orthonormal base b_1, b_2,...,b_N of $\mathbb{R}^N$ such that $\{b_1, b_2, b_3\}$ span $\mathcal{M}$, with b_3 pointing in the directon of the k_3-gradient. We assume that $\mathbf{t}$=$\mathcal{F}$(t') is a regular point of $\mathcal{M}$, meaning that the partial derivatives of $\mathcal{F}$ in t' are linear independent. This allows constructing the base from the partial

[1] Since Δk_3 averages out to zero by the way we designed the noise.

derivatives by a procedure known as Gram Schmidt orthonormalization. It begins with

$$\begin{aligned} v_1 &:= \frac{\partial \mathcal{F}}{\partial q_1}(t') \\ b_1 &:= \frac{v_1}{\|v_1\|} \\ v_2 &:= \frac{\partial \mathcal{F}}{\partial k_2}(t') - \left\langle \frac{\partial \mathcal{F}}{\partial k_2}(t'), b_1 \right\rangle b_1 \\ b_2 &:= \frac{v_2}{\|v_2\|} \\ v_3 &:= \frac{\partial \mathcal{F}}{\partial k_3}(t') - \left\langle \frac{\partial \mathcal{F}}{\partial k_3}(t'), b_2 \right\rangle b_2 - \left\langle \frac{\partial \mathcal{F}}{\partial k_3}(t'), b_1 \right\rangle b_1 \qquad \text{(B.3)} \\ b_3 &:= \frac{v_3}{\|v_3\|} \end{aligned}$$

b_4,...,b_N can be arbitrarily chosen so as to fill up $\{b_1, b_2, b_3\}$ to an orthonormal base of $\mathbb{R}^N$.

B.2.1 Computing the k_3-gradient

Consider y=t'=(q_1,k_2,k_3), so $\mathcal{F}(y)$=$\mathbf{t}$. If its k_3 is modified, $\mathcal{F}(y)$ moves in the direction of

$$\frac{\partial \mathcal{F}}{\partial k_3}(t') \qquad \text{(B.4)}$$

at speed $\left\|\frac{\partial \mathcal{F}}{\partial k_3}(t')\right\|$. Inverting the situation by moving $\mathcal{F}(y)$ in that direction, results in k_3 changing by

$$\frac{1}{\left\|\frac{\partial \mathcal{F}}{\partial k_3}(t')\right\|} \qquad \text{(B.5)}$$

per unit of movement.

Now the k_3-gradient is orthogonal to the plane of constant k_3 within $\mathcal{M}$, that plane is spanned by $\frac{\partial \mathcal{F}}{\partial k_2}(t')$ and $\frac{\partial \mathcal{F}}{\partial q_1}(t')$, and therefore also by b_1 and b_2. Hence the direction of the k_3-gradient is obtained by subtracting from $\frac{\partial \mathcal{F}}{\partial k_3}(t')$ its components in b_1 and b_2 direction. So it happens to be v_3 of equation (B.3). By construction, $v_3(t')$ is shorter than $\frac{\partial \mathcal{F}}{\partial k_3}(t')$. The subtracted components lie in a plane where k_3 is constant, so $\mathbf{t}$+$v_3(t')$ and $\mathbf{t}$+$\frac{\partial \mathcal{F}}{\partial k_3}(t')$ have the same k_3. Therefore, in the direction of v_3, k_3 changes faster than in the direction of the partial derivative, by a factor of

$$\frac{\left\|\frac{\partial \mathcal{F}}{\partial k_3}(t')\right\|}{\|v_3(t')\|} \qquad \text{(B.6)}$$

Take this factor times (B.5) to get the speed of k_3 change in the direction of v_3, the result is $\frac{1}{\|v_3\|}$. Hence the k_3-gradient is given by

$$k_3 \ gradient = \frac{1}{\|v_3\|} \cdot b_3 \tag{B.7}$$

.

B.2.2 Modeling anisotropic Noise

$\frac{1}{N-3}$ of equation (B.2) is a conversion factor translating from the mean squared noise component **o**rthogonal to $\mathcal{M}$ to the mean squared noise component in direction of the k_3-**g**radient. In the isotropic situation it could be assumed constant. Now it needs to be replaced with a function *ogconvert*(t'), which accounts for

- the shape of the noise cloud at $\mathbf{t}$=$\mathcal{F}(t')$
- the inclination of $\mathcal{M}$ as it intersects with that cloud (see Figure B.1).

Equation (B.2) thus becomes in the anisotropic case

$$k3var = \frac{\|\vec{\mathbf{fa}}\|^2}{\|v_3\|^2} \cdot ogconvert(t') \tag{B.8}$$

where we substituted $\frac{1}{\|v_3\|}$ for $\|k_3 \ gradient\|$ because of equation (B.7).

Now what is the real shape of the noise cloud? Every frame is affected independently by Gaussian noise. If all frames had the same amount of noise, the distribution would be spherical. If the amount is not equal, we get an ellipsoid whose main axes are the coordinate axes. (A skew ellipsoid would imply dependency between the frames, which can be safely ruled out.)

Let s_i be the standard deviation of the noise of frame i. Together, the frames define a **noise vector x** = $(x_1,\dots,x_n)$. For i=1 to N, x_i is normally distributed with mean 0 and standard deviation s_i. We have to compute the mean squared projections of x on the orthocomplement of $\mathcal{M}$, to be called $\mathbf{msp_{ortho}}$, and on the k_3-gradient, $\mathbf{msp_{grad}}$. Then

$$ogconvert = \frac{msp_{grad}}{msp_{ortho}} \tag{B.9}$$

and we are done.

We use the metric of $\mathbb{R}^N$ defined by the inner product (4.7). The unit vectors with respect to this metric are

$$e_i \ = \ (0,\dots,0,\frac{1}{\sqrt{w_i}},0,\dots,0) \tag{B.10}$$

where w_i are the Decay Weights (2.7). After conversion to units of this metric, the components x_i of x have therefore standard deviations $s_i\sqrt{w_i}$.

$\mathcal{M}$ cuts skewly through that cloud of noise. We use the orthonormal base $\{b_1, b_2, \ldots, b_N\}$ defined at the beginning, whose b_1, b_2, b_3 are spanning $\mathcal{M}$, $b_4, \ldots, b_N$ are spanning its orthocomplement and b_3 points in the direction of the k_3-gradient. Let, for j=1,...,N, $\mathbf{msp_j}$ denote the mean squared length of the projection of $\mathbf{x}$ onto b_j. It happens to be also the variance of the signed length of that projection, this will facilitate its computation. Consider one component x_i of the noise vector. It points in direction e_i. The projection of e_i on b_j is of length $0 \leq |\langle e_i, b_j \rangle| \leq 1$. As x_i is projected onto b_j, its length gets multiplied with that factor. The projection remains Gaussian and its standard deviation is $s_i \cdot \sqrt{w_i} \cdot |\langle e_i, b_j \rangle|$. The variance is obtained by squaring:

$$variance = s_i^2 \cdot w_i \cdot \langle e_i, b_j \rangle^2 \tag{B.11}$$

Let $b_{j,i}$ denote the i^{th} component of b_j. Then the inner product can be computed from (4.7) and (B.10):

$$\langle e_i, b_j \rangle = \sqrt{w_i} \cdot b_{j,i}$$

Plugging this into the variance (B.11) and summing up over all components of $\mathbf{x}$=$(x_1, \ldots, x_N)$ gives the desired result:

$$msp_j = \sum_{i=1}^{N} s_i^2 \cdot b_{j,i}^2 \cdot w_i^2 \tag{B.12}$$

Here summation is allowed because we are adding independent random variables: the projections of the components of x, which are associated with the scanframes, onto b_j. Because of independence, the sum of variances is the variance of the sum. Besides, the distribution is still normal.

$$msp_{grad} = msp_3$$

is a special case. msp_{ortho} is obtained by summation:

$$msp_{ortho} = \sum_{j=4}^{N} msp_j = \sum_{j=4}^{N} \sum_{i=1}^{N} s_i^2 \cdot b_{j,i}^2 \cdot w_i^2 \tag{B.13}$$

Here, summation is justified by an argument involving the Fourier Decomposition (see B.4).

Now, using equation (B.13) to compute msp_{ortho} would require providing b_4 to b_N for every voxel, which is an unreasonable effort. It is better to exploit the relation

$$\underbrace{\sum_{j=4}^{N} \sum_{i=1}^{N} s_i^2 \cdot b_{j,i}^2 \cdot w_i^2}_{=msp_{ortho}} = \underbrace{\sum_{j=1}^{N} \sum_{i=1}^{N} s_i^2 \cdot b_{j,i}^2 \cdot w_i^2}_{=:msl_x} - \underbrace{\sum_{j=1}^{3} \sum_{i=1}^{N} s_i^2 \cdot b_{j,i}^2 \cdot w_i^2}_{=:msp_{\mathcal{M}}} \tag{B.14}$$

where $\mathbf{msp}_{\mathcal{M}}$ is cheaper to compute, and $\mathbf{msl_x}$ is the mean squared length of the noise vector x:

$$msl_x = \sum_{j=1}^{N} \sum_{i=1}^{N} s_i^2 \cdot b_{j,i}^2 \cdot w_i^2 \tag{B.15}$$

As the special choice of the orthonormal basis b_i has not been used in the path to equation (B.15), we can replace it by any orthonormal base. In particular, the result stays the same if the e_j are used instead of the b_j:

$$msl_x = \sum_{j=1}^{N} \sum_{i=1}^{N} s_i^2 \cdot e_{j,i}^2 \cdot w_i^2$$

But since $e_{j,i}$ vanishes for $i \neq j$ the double sum collapses to a single one:

$$msl_x = \sum_{j=1}^{N} s_j^2 \cdot e_{j,j}^2 \cdot w_j^2 = \sum_{j=1}^{N} s_j^2 \cdot w_j \tag{B.16}$$

since $e_{j,j} = \frac{1}{\sqrt{w_j}}$. Using this and equation (B.14), (B.9) becomes

$$ogconvert = \frac{msp_{grad}}{msl_x - msp_{\mathcal{M}}} = \frac{\sum\limits_{i=1}^{N} s_i^2 \cdot b_{3,i}^2 \cdot w_i^2}{\sum\limits_{i=1}^{N} s_i^2 \cdot w_i - \sum\limits_{j=1}^{3} \sum\limits_{i=1}^{N} s_i^2 \cdot b_{j,i}^2 \cdot w_i^2} \tag{B.17}$$

B.2.3 The Shape of the Noise Cloud

It remains to estimate the standard deviations s_i of the noise vector components. When assuming Decay Proportional Noise (see section 2.7.1) for the frames, it translates to isotropic noise as is demonstrated in section 2.7.4. So in this case we could spare the effort of section B.2.2 and use the simplified versions (B.2) and (8.3) of the formula.

Using the anisotropic version of the formula, the estimate can be adapted to every voxel by factoring in its TAC C_t, assuming Proportional Noise (section 2.5) for the raw signal. $C_t(i)$ are the intensities on the reconstructed frames, after the reconstruction software has applied Combined Correction (section 2.6.1), which is division by the w_i (section 2.7). So the raw signal was proportional to $w_i \cdot C_t(i)$, and its SD proportional to $\sqrt{w_i \cdot C_t(i)}$. That leads to $SD \sim \sqrt{\frac{C_t(i)}{w_i}}$ after Combined Correction. So we use

$$s_i := \sqrt{\frac{C_t(i)}{w_i}} \tag{B.18}$$

for computation of *ogconvert.* It is safe to ignore the constant of proportionality since formula (B.17) is indifferent to scaling of the s_i.

B.2.4 Application to real and Simulator Data

In its version (B.8) the estimate depends on the triple t' of true kinetic constants which is unknown in the real situation. Hence we must replace all its occurrences by the closest available approximation f'. This also affects computation of b_1, b_2, b_3 and v_3 and leads to the final version (8.2), which we use both for simulations and real PET data.

B.3 Limitations

Which simplifications in the design of *k3var* might lead to bad error estimates?

1. Linearization. $\mathcal{M}$ has been replaced by its tangential space in **t**. So its nonlinear nature might cause errors especially in areas of strong curvature, notably close to the reference curve and its scalar multiples which are singular points of $\mathcal{M}$. If we reduce the amount of simulated noise, it downsizes the geometry of the noise cloud, making everything smaller compared to $\mathcal{M}$. On a magnified view, $\mathcal{M}$ comes closer to being linear. This should improve precision of *k3var*.

2. Replacing **t'** by **f'**. This induces error arising from the difference netween **f** and **t**. Like the previous, this type of error should go away when we simulate less noise, since it brings **f** and **t** closer together.

3. Ignoring noise of C_r. While the C_t correspond to points on $\mathcal{M}$, C_r is part of the very definition of $\mathcal{F}$ and $\mathcal{M}$. Its noise will therefore produce a "noisy" version of $\mathcal{M}$ which is harder to approximate by simulated C_t. By this mechanism, *k3var* should be sensitive to noise of C_r. But the extent of this effect is unknown since we made no effort to quantify it. It should respond selectively to increasing the reference volume, which reduces the noise of C_r.

B.4 Adding up Mean Squared Projections

Given an orthonormal base b_1,…,b_N relative to some inner product $\langle\cdot|\cdot\rangle$, every vector x can be written in its "Fourier Decomposition" form:

$$x = \sum_{j=1}^{N} \langle x|b_j\rangle b_j$$

Now consider a subset of the b_j, for simplicity, $b_1, \ldots, b_k$. Then the partial sum

$$x_p := \sum_{j=1}^{k} \langle x|b_j\rangle b_j$$

is the projection of x on the subspace S spanned by b_1,...,b_k. Its squared length can hence be computed:

$$\langle x_p | x_p \rangle = \sum_{j=1}^{k} \langle x | b_j \rangle^2$$

and the summands $\langle x | b_j \rangle^2$ happen to be the squared lengths of the projections of x on the b_i. Hence these squared lengths add up to the squared projection on S. Now apply this equation to a large number of vectors $x_1, \ldots, x_M$:

$$\sum_{i=1}^{M} \langle x_{ip} | x_{ip} \rangle = \sum_{i=1}^{M} \sum_{j=1}^{k} \langle x_i | b_j \rangle^2$$

Then exchanging the order of summation on the right side and dividing by M leads to

$$\frac{1}{M} \sum_{i=1}^{M} \langle x_{ip} | x_{ip} \rangle = \sum_{j=1}^{k} \frac{1}{M} \sum_{i=1}^{M} \langle x_i | b_j \rangle^2$$

So the mean squared projections to the one dimensional subspaces add up to the mean squared projection on S, this has been used to obtain equation (B.13).

Bibliography

[1] Arai H, Kosaka K, Muramoto O, Moroji T, Iizuka R: “A biochemical study of cholinergic neurons in the post-mortem brains from the patients with Alzheimer-type dementia”, Clin Neurol(Tokyo) 24 (1984): 1128-1135

[2] Aston JAD, Cunningham VJ, Asselin MC, Hammers A, Evans AC, Gunn RN: “Positron Emission Tomography Partial Volume Correction: Estimation and Algorithms”, J Cereb Blood Flow Metab. 2002; 22: 1019-34

[3] Atack JR, Perry EK, Bonham JR, Candy JM, Perry RH: “Molecular forms of acetylcholinesterase and butyrylcholinesterase in the aged human central nervous system”, J Neurochem 47 (1986): 263-277

[4] Bartzokis G, Lu PH, Mintz J: “Human brain myelination and amyloid beta deposition in Alzheimer’s disease”, Alzheimer’s and Dementia, 2007, 3: 122-125

[5] Blomqvist G: “On the construction of functional maps in positron emission tomography”, J Cereb Blood Flow Metab. 1984; 4: 629-632

[6] Brett M, Johnsrude IS, Owen AM: “The problem of functional localization in the human brain”, Nature Reviews: Neuroscience, 2002, 3: 243-249

[7] Cizek J, PhD thesis, “Robust Algorithms for Registration of 3D Images of Human Brain”, University of Cologne, 2005

[8] Cizek J, Herholz K, Vollmar S, Klein JC, Schrader R, Heiss WD: “Fast and Robust Registration of PET and MR images of human brain”, Neuroimage 2004, 22(1): 434-442

[9] Dennis JE, Schnabel RB: “Numerical Methods for Unconstrained Optimization and Nonlinear Equations”, Prentice-Hall, Englewood Cliffs, New Jersey, 1983

[10] Fletcher R: “Practical Methods of Optimization”, Vol.1, John Wiley and Sons, New York, 1980

[11] Florea I, PhD thesis, “PET Parametric Imaging of Acetylcholinesterase Activity without arterial blood sampling in normal Subjects and Patients with neurodegenerative Disease”, University of Padua, 2008

[12] Gunn R, Gunn S, Turkheimer F, Aston J, Cunningham V: “Positron Emission Tomography Compartmental Models: A Basis Pursuit Strategy for Kinetic Modelling”, J Cereb Blood Flow Metab. 2002, 22(12): 1425-1439

[13] Herholz K, private communication

[14] Herholz K, Weisenbach S, Zuendorf G, Lenz O, Schroeder H, Bauer B, Kalbe E, Heiss WD: “In vivo study of acetylcholine esterase in basal forebrain, amygdala, and cortex in mild to moderate Alzheimer disease”, Neuroimage 2004, 21(1): 136-43

[15] Herholz K, Lercher M, Wienhard K, Bauer B, Lenz O, Heiss WD: “PET measurement of cerebral acetylcholine esterase activity without blood sampling”, Eur J Nucl Med. 2001, 28(4): 472-77

[16] Herholz K, Herscovitch P, Heiss WD: “Neuro PET”, Springer Verlag, Berlin Heidelberg, 2004

[17] Herholz K, Bauer B, Wienhard K, Kracht L, Mielke R, Lenz O, Strotmann T, Heiss WD: “In vivo measurements of regional acetylcholine esterase activity in degenerative dementia: comparison with blood flow and glucose metabolism”, J Neural Transm 2000, 12: 1457-1468

[18] Ikoma Y, Watabe H, Shidahara M, Naganawa M, Kimura Y: “PET kinetic analysis: error consideration of quantitative analysis in dynamic studies”, Ann Nucl Med. 2008, 22: 1-11

[19] Irie T, Fukushi K, Akimoto Y, Tamagami H, Nozaki T: “Design and evaluation of radioactive acetylcholine analogs for mapping brain acetylcholinesterase (AChE) in vivo”, Nucl Med Biol. 1994, 21(6): 801-808

[20] Irie T, Fukushi K, Namba H, Iyo M, Tamagami H, Nagatsuka S, Ikota N: “Brain acetylcholinesterase activity: validation of a PET tracer in a rat model of Alzheimer’s disease”, J Nucl Med. 1996, 37(4): 649-655

[21] Kilbourn MR, Nguyen TB, Snyder SE, Sherman P: “N-[11C]mythylpiperidine esters as acetylcholinesterase substrates: an in vivo structure-reactivity study”, Nucl Med Biol. 1998, 25(8): 755-760

[22] Knuth DE: “The Art of Computer Programming”, section 3.4.1, third edition (1998), Addison Wesley

[23] Koeppe RA, Frey KA, Snyder SE, Meyer P, Kilbourne MR, Kuhl DE: “Kinetic modeling of N-[^{11}C]methylpiperidin-4-yl propionate: alternatives for analysis of an irreversible PET tracer for measurement of acetylcholine esterase activity in human brain”, J Cereb Blood Flow Metab. 1999, 19: 1150-1163

[24] Krais R, private communication

[25] Krais R, seminar on PET at MPIFNF

[26] Laboratory of Neuro Imaging, UCLA, Online Resource: http://www.loni.ucla.edu

[27] Marquardt D: “An algorithm for Least-Squares Estimation of Nonlinear Parameters”, SIAM Journal on Applied Mathematics 11: 431-441

[28] The MathWorks Inc., Online Resource: “MATLAB is a high-level language and interactive environment”, http://www.mathworks.com/products/matlab/

[29] Nagatsuka S, Fukushi K, Shinotoh H, Namba H, et al.: “Kinetic analysis of [^{11}C]MP4A using a high-radioactivity brain region that represents an integrated input function for measurement of cerebral acetylcholinesterase activity without arterial blood sampling”, JCBFM (2001), 21: 1354-1366

[30] Namba H, Fukushi K, Nagatsuka S, Iyo M, Shinotoh H, Tanada S, Irie T: "Positron emission tomography: quantitative measurement of brain acetylcholinesterase activity using radiolabeled substrates", Methods Jul 2002, 27(3): 242-250

[31] Namba H, Iyo M, Fukushi K, Shinotoh H, Nagatsuka S, Suhara T, Sudo Y, Suzuki K, Irie T: "Human cerebral acetylcholine esterase activity measured with positron emission tomography: procedure, normal values and effect of age", Eur J Nucl Med. 1999, 26: 135-143

[32] Nokia Incl, Online Resource: "Qt - A cross-platform application and UI framework", http://qt.nokia.com/

[33] Reinikainen KJ, Riekkinen PJ, Paljarvi L, Soininen H, Helkala EL, Jolkkonen J, Laakso M: "Cholinergic deficit in Alzheimer's disease: a study based on CSF and autopsy data", Neurochem Res 1988; 13:135-146

[34] Singh M, Kanji GK: "Fitting a non-linear model with errors in both variables and its application", Journal of Applied Statistics, 1988, 15(3): 267-274

[35] SPM, Online Resource: "The SPM software package has been designed for the analysis of brain imaging data sequences", http://www.fil.ion.ucl.ac.uk/spm/

[36] Tanaka N, Fukushi K, Shinotoh H, Nagatsuka S, Namba H, Masaomi I, Aotsuka A, Ota T, Tanada S, Irie T: "Positron Emission Tomographic Measurement of Brain Acetylcholinesterase Activity using N-[^{11}C]methylpiperidin-4-yl Acetate without arterial Blood Sampling: Methodology of Shape Analysis and its Diagnostic Power for Alzheimer's disease", J Cereb Blood Flow Metab. 2001, 21(3): 295-306

[37] Varrone A, private communication

[38] Vollmar S, PhD thesis, "Integration of Functional Imaging (PET) to Surgery Planning in Brain Tumors", University of Cologne, 2006

[39] Vollmar S, Hampl J, Kracht L, Herholz K: "Integration of Functional Data (PET) in Brain Surgery Planning and Neuronavigation", Advances in Medical Engineering, Springer Proceedings in Physics 114 (2007): 98-103

[40] Vollmar S, Sué M, Krais R, Hohmann C, Huesgen A: "Vinci 3 and experiences with Multi-Platform Development", Abstracts of the XI Turku PET Symposium (2008): p. 104

[41] Vollmar S, Sué M, Huesgen A, Hohmann C, Online Resource: "VINCI Homepage", http://www.nf.mpg.de/vinci, 2009

[42] Wienhard K, private communication

[43] Wienhard K, Wagner R, Heiss WD: "PET: Grundlagen und Anwendungen der Positronen-Emissions-Tomographie", Springer Verlag, Berlin Heidelberg, 1989

[44] Wikipedia, The free Encyclopedia. Online Resource, en.wikipedia.org

[45] Zuendorf G, private communication

[46] Zuendorf G et al.: "PET functional parametric Images of Acetylcholine Esterase Activity without Blood Sampling" in: Senda M et al (eds), Brain Imaging Using PET, Academic Press, San Diego CA, USA, pp 41-46, Proceedings for BrainPET 2001

List of Figures

List of Tables

Glossary

This contains terms and symbols of global use. Whatever is not here may be defined in the chapter where you found it.

a	measured TAC C_t, section B.1, Figures 4.2 and 8.1
abba-a (-b, -t)	section 14.1.4
a,b-comparison	sections 14.1.4, 14.1.9
AChE	Acetylcholine Esterase, section 1.3
acquisition	section 1.1.1
Acquisition Space	section 9.2
activity	section 2.1
activity (of AChE)	section 1.5
AD	Alzheimer's Disease, section 1.2
Affine Normalization	section 2.10
Assumption (1,. . . ,3)	chapter 3
Assumption 4	section 3.2.2
atlas	sections 9.1, 10.1
attenuation	section 1.1
Becquerel	section 2.1
Blood Input Function	C_{pl}, section 3.1
Bq	Becquerel, section 2.1
Cbl	Cerebellum
Cerebellum 3500-2	section 10.3.1
Cerebellum Decay Fraction	section 7.1.3
C_i	section 3.2.2, formula (3.6)
coincidence	section 1.1.1
COLOGNE	sections 1.7, 3.3 ff.
Combined Correction	section 2.6.1
Combined Weighting	section 7.3
coregister	section 2.10
'corr'	section 2.8
Cortex Digest	section 14.1.2
COV	Coefficient of Variation, section 2.4
C_{pl}	Blood Input Function, section 3.1

C_r	reference curve, section 3.2.2, formula (3.5)
C_r Block Simulation	section 7.2.2
C_r Weighting	section 7.3
C_t	section 3.1
CT	Computed (X-ray-) Tomography
Decay Correction	section 2.6
Decay Fraction	section 7.1.3
Decay Function	section 2.6
Decay Proportional Noise	section 2.7.1
Decay Weighting	section 2.7
Dilation	section 10.2
DiMI	Diagnostic Molecular Imaging, section 2.11
Discretization Bias	section 6.5
‘dist’	section 2.8, formula (2.10)
Double Random Simulation	section 7.2.1
dynamic PET	section 1.1.5
Early Frames	section 9.2.1
ECAT7	section 2.2
empiric quotient	section 7.1.4.2
equivalent volume	section 7.1.5
Erosion	section 10.2
Evaluation Format	image format of Evaluation Space, section 9.2
Evaluation Space	section 9.2
$\mathcal{F}$	section 4.1, formula (4.11)
f	$\mathcal{F}(f')$, foot of perpendicular, Figures 4.2, 8.1
f’	triple found by NLS, section 4.1, Figures 4.2, 8.1
failrate	100% minus yield of section 10.5
failselection	section 10.4
FDG	section 1.1.2
filter kernel	section 2.9
filter mask	section 10.4.3
Fixed C_r Simulation	section 7.2.3
flow	perfusion, section 9.2
Flow Image	section 9.2.1
frame	section 1.1.5
framedev	section 4.7.3
framing (-schedule)	section 6.1.1
full brain mask	section 10.1
full mask	section 14.1.5
FWHM	width of Gauss Filter, section 2.9
FWHM2	section 10.3
FWHM3	section 10.4.3
Gauss Filter	section 2.9
High Intensity Zones	section 10.6.1

Hippo	Hippocampus
Hippocampus Digest	section 14.1.2
ideal reference region	satisfies Assumption 4 of section 3.2.2
Infinity Image	section 7.1.3.1
Input Function	C_{pl}, section 3.1
intensity	section 2.2
Intensity, Overall	section 10.2
Interfile	section 2.2
Interpolation Bias	section 6.3
Iso framings	section 6.1.1.2
isotropic noise	sections 2.7.4, B.1
iter	section 4.7.5
k	k_2+k_3, sections 3.1, 4.5.1
k_1,k_2,k_3	sections 1.5 and 3.1, Figure 1.2
k_{1r},k_{2r},k_{3r}	k_1,k_2,k_3 of the reference region
k_3-digest	section 14.1.2
k_3-gradient	section B.1
k3dev	section 8.1
k3diff	section 4.7.6
k_{3r}-correction	section 3.2.3
k_{3r}-induced bias	section 6.4
k3var	section 8.1
k_r	k_{2r}+k_{3r}
l_1, l_2	section 5.3
λ	section 5.4
Late Frames	section 9.2.1
Line of Response	section 1.1
LOR	Line of Response, section 1.1
Low Intensity Zones	section 10.6.1
L,R-comparison	sections 14.1.5, 14.1.9
$\mathcal{M}$	section 4.2
‘margin’	section 10.4.3
mask	section 2.2
masked filtered frame	section 10.4.3
MBq	Megabecquerel, section 2.1
MCI	Mild Cognitive Impairment, section 2.11
mCi	milliCurie
Medium Zones	section 10.6.1
modality (of PET)	section 2.10
modality (of parametric image)	section 4.7
Model Compliant	belonging to $\mathcal{M}$, section 4.2
Model Curve	section 4.2
Model Function	$\mathcal{F}$, formula (4.11)

MP4A	section 1.4
MP4P	section 1.4
MPIFNF	Max Planck Institute for Neurological Research, Cologne
MR	Magnetic Resonance image
MRI	Magnetic Resonance Imaging
Mutual Information	section 2.8
n_1	section 5.3
NLS	Nonlinear Least Squares, chapter 4
$noise_{abs}$	section 4.7.1
Noise-induced bias	section 7.6
$noise_{rel}$	section 4.7.2
normalize	section 2.10
Normalization Template	section 9.2
ogconvert	section B.2.2
Overall Intensity	section 10.2
‘padding’	section 10.3
Partial Volume Effects	section 1.1.4.2
perfusion	section 9.2
PET	section 1.1
p.i.	post injection (of tracer)
postselection	section 10.4
preselection	section 10.4
Privileged Area	section 10.5.1
Privileged Intersection Set	section 10.6
Proportional Noise	section 2.5
Put	Putamen
Putamen 674+2	section 10.3.1
Putamen Decay Fraction	section 7.1.3
P-value	section 2.4
PVE	Partial Volume Effects, section 1.1.4.2
q_1	k_1/k_{1r}, section 4.1
‘qzoning’	section 10.4.3
reconstruction	section 1.1.1
reference curve	C_r, section 9.4, chapters 11 and 14
reference mask	section 9.4
reference region	sections 3.2.2, 9.4
‘regionsize’	section 10.3
regular point of $\mathcal{M}$	section 4.2
Rigid Body Coregistration	section 2.10
ROI	region of interest, section 2.2
S_1	universal starting point, section 5.2
S_2,. . . ,S_5	starting points used in chapter 5 and A.3
S3 Input Function	sections 6.1.2.1, 7.1.1

Ich versichere, dass ich die von mir vorgelegte Dissertation selbständig angefertigt, die benutzten Quellen und Hilfsmittel vollständig angegeben und die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen - die anderen Werken im Wortlaut oder dem Sinn nach entnommen sind, in jedem Einzelfall als Entlehnung kenntlich gemacht habe; dass diese Dissertation noch keiner anderen Fakultät oder Universität zur Prüfung vorgelegen hat; dass sie - abgesehen von unten angegebenen Teilpublikationen - noch nicht veröffentlicht worden ist sowie, dass ich eine solche Veröffentlichung vor Abschluss des Promotionsverfahrens nicht vornehmen werde. Die Bestimmungen der Promotionsordnung sind mir bekannt. Die von mir vorgelegte Dissertation ist von Herrn Prof. Dr. Rainer Schrader betreut worden.

Christoph Hohmann

Klein JC, Eggers C, Kalbe E, Thomas A, Weisenbach S, Hohmann C, Vollmar S, Baudrexel S, Diederich NJ, Heiss WD, Hilker R: *"Neurotransmitter profiles in Dementia with Lewy Bodies and Parkinson's disease dementia"*. Submitted to: Neurology (2009)

Haense C, Kalbe E, Hohmann C, Krais R, Neumaier B, Herholz K, Heiss WD: *"Memory and cholinergic system function in Mild Cognitive Impairment"*. In preparation (2009)