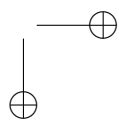
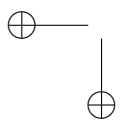
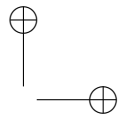
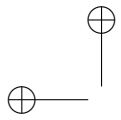


# Tomographic Reconstruction using Anatomy and Regularization



**IMM**

**Esben Høgh-Rasmussen**

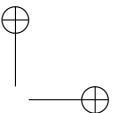
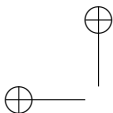


# Tomographic Reconstruction using Anatomy and Regularization



IMM

Esben Høgh-Rasmussen



**Title** Tomographic Reconstruction using Anatomy and Regularization  
**Author** Esben Høgh-Rasmussen  
**Keywords** TOMOGRAPHIC RECONSTRUCTION  
LARGE-SCALE COMPUTATIONS  
KRYLOV, SVD, LANCZOS, BIDIAGONALIZATION.

**Supervisors** Lars Kai Hansen, IMM  
Per Christian Hansen, IMM  
Claus Svarer, NRU

**Typesetting** L<sup>A</sup>T<sub>E</sub>X and B<sup>I</sup>B<sub>T</sub>E<sub>X</sub>  
Main text: Times-Roman  
Mathematics: Computer-Modern  
Headlines and notes: Helvetica

**Graphs** MATLAB<sup>®</sup>



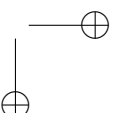
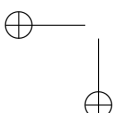
Technical University of Denmark  
Informatics and Mathematical Modelling  
Building 321, DK-2800 Kongens Lyngby, Denmark  
Phone (+45) 45253351, Fax (+45) 45882673  
reception@imm.dtu.dk  
www.imm.dtu.dk



Neurobiology Research Unit  
Department of Neurology  
The Neuroscience Centre, Rigshospitalet  
Blegdamsvej 9, Copenhagen University Hospital  
Building 9201, DK-2100 Copenhagen, Denmark  
Phone (+45) 35456711, Fax (+45) 35456713  
www.nru.dk

Copyright © 2005, Esben Høgh-Rasmussen.

**Reference** IMM-PHD-2005-156  
**ISSN** Unassigned  
**ISBN** Unassigned



# Abstract

## Abstract (English)

**Title:** **Tomographic Reconstruction using Anatomy and Regularization.**

This thesis develops a method for reconstructing images in emission tomography. Rooted in user requirements, it is decided to go forward with Tikhonov regularization followed by post-filtering with a Gaussian kernel. This is followed by an analysis of the theoretical loss of performance implied by linear methods.

After this, the focus switches to development of algorithms for large-scale SVD-problems and regularized solutions of linear systems. This culminates in a toolbox for MATLAB<sup>®</sup>, BBTools, which implements these algorithms in a familiar and practical setting.

Finally, we show the advantages of the method by comparing several images reconstructed using both the new and a classical algorithm. We also show that anatomical structures can be incorporated by means of restrictions.

## Resumé (Danish)

**Titel:** **Tomografisk rekonstruktion med brug af anatomi og regularisering.**

Denne afhandling udvikler en metode til rekonstruktion af billeder inden for emissionstomografi. Med udgangspunkt i brugerkrav besluttet det at fortsætte med Tikhonov regularisering efterfulgt af en filtrering med en Gausskerne. Dette følges op med en analyse af de teoretiske ulemper ved lineære metoder.

Herefter skifter fokus til at udvikle algoritmer, der kan håndtere beregninger af store SVD-problemer og bestemme regulariserede løsninger af lineære systemer. Dette kulminerer i en pakke til MATLAB<sup>®</sup>, BBTools, der implementerer disse algoritmer i et velkendt og anvendeligt miljø.

Endeligt demonstreres fordelene ved den nye metode ved at sammenligne et antal billeder, der er rekonstrueret med både den nye metode og en klassisk metode. Det vises at anatomiske strukturer kan respekteres gennem restriktioner.

# Preface

This thesis was prepared at the Neurobiology Research Unit (NRU), Copenhagen University Hospital, and Informatics and Mathematical Modeling (IMM), Technical University of Denmark, in partial fulfillment of the requirements for acquiring the PhD degree in engineering.

The aim of the work is to improve the quality of the images produced in emission tomography, i.e. PET (Positron Emission Tomography) and SPECT (Single Photon Emission Computed Tomography). Improved image quality would have wide-spread diagnostic applications, both in medical research and clinically. For instance, better images would facilitate diagnosis of cancer, Alzheimer's disease and its early stage known as MCI (mild cognitive impairment), epilepsy, and many others.

In medicine, almost all images are produced using one of two algorithms: Filtered Back-Projection (FBP) or Expectation Maximization (EM). They have been used for many years, although relatively primitive.

Many technologically oriented minds have stumbled upon this fact. I am by no means the first to have attempted to make a better solution; indeed it appears to be a frequently presented topic for a PhD work. Many of these projects have successfully shown that they can produce better images than either of the two classical methods.

So why do images continue to be produced using these two algorithms?

One major reason is that medical usage have a number of requirements that are difficult, or impossible, to quantify. These requirements tends to be ignored, because it is discouraging to be unable to measure whether a progress really has been made.

If the success of a method is measured by the number of doctors using a new method after, say, 10 years, then the goals to be pursued changes radically. One example is backwards compatibility, i.e. the ability to imitate the previous methods. So far only the EM algorithm have been able to overcome this obstacle. Unlike previous projects, I have devoted a chapter to describe these requirements.

This project is an attempt to make evolutionary, rather than revolutionary, progress in image formation. There are a number of reasons that made me believe I could succeed when I started this PhD.

Ignorance

- When I started it seemed easy to improve on the current methods. This was only true because I was naive and unaware of prior art.

Wrong assumptions

- The method chosen for the computational work is a text-book solution used in many scientific fields. I therefore assumed that off-the-shelf software was available to tackle the problem. As it turned out, this was not the case

	and, consequently, an excessive fraction of my time as a PhD student had to be devoted to software development.
Computational power	<ul style="list-style-type: none"> <li>• The power of computers increases at a breath-taking pace. A problem as important as the one addressed in this thesis, which relies heavily on computations, should continuously try to take advantage of this.</li> </ul>
Engineering spirit	<ul style="list-style-type: none"> <li>• As an engineer, I have been trained to seek solutions that will work for the intended user. This is, in my opinion, the weakest point in most previous work and the real reason why many previous attempts to improve currently used methods have failed to be adopted clinically.</li> </ul>
User knowledge	<ul style="list-style-type: none"> <li>• I have spent a great deal of time at NRU listening to doctors who use the images. Even though they are top researchers in their fields, hence willing to adopt new methods faster than most medical staff, they have comparable needs.</li> </ul>

## Acknowledgments

First, I wish to thank the organizations and private funds who funded this research:

- EU 5th Framework Programme (QLK6-CT-2000-00502, NCI-MCI) Neuroreceptor Changes in Mild Cognitive Impairment
- EU 6th Framework Programme (LSHB-CT-2005-512146, DiMI) Diagnostic Molecular Imaging
- H:S – Copenhagen Hospital Corporation
- M.L. Jørgensens og Gunnar Hansens Fond (J. nr. 2956)
- Savværksejer Jeppe Juhl og hustru Ovita Juhls Mindelegat
- Otto Mønstedts Fond (J. nr. 03-71-898)

I would also like to thank Misha Kilmer, whom I visited 6 months at Tuft's University in Boston, USA. She providing a great large-scale test-problem in diffusion optical tomography. Such test-problems are invaluable when developing numerical software. I also thank her for pointing me at [53] which includes a proof that is fundamental to the computation routines in chapter 7.

Furthermore, I thank my brother, Nis Høgh-Ramussen, for encourage- and discouragement throughout, for helping reduce a 3D-problem to 2D in chapter 6, and the maxim "if you can not express what you want, then you will not get it".

Finally, I thank NRU and everyone there for supporting the project, not least financially. In particular, everyone there has been very patient during the long periods where I have been buried in a dark mathematical pit, where only I could (occasionally) glimpse where the project was heading.

Copenhagen, September, 2005

---

Esben Høgh-Rasmussen

# Contents

<b>Abstract</b> . . . . .	<b>iii</b>
Abstract (English) . . . . .	iii
Resumé (Danish) . . . . .	iii
<b>Preface</b> . . . . .	<b>iv</b>
Acknowledgments . . . . .	v

## Part I: Propaedeutic

<b>1 Introduction</b> . . . . .	<b>3</b>
1.1 Tomographic reconstruction . . . . .	3
1.2 Thesis outline . . . . .	3
1.3 On the writing style . . . . .	4
1.4 Coordinates and directions . . . . .	4
1.5 Linear algebra . . . . .	5
1.6 Software and programs . . . . .	6
1.7 Literate programming . . . . .	6
1.8 Numerical notation . . . . .	7
<b>2 Scanner instrumentation</b> . . . . .	<b>9</b>
2.1 Scanners and their abbreviations . . . . .	9
2.2 Resolution . . . . .	10
2.3 The CT scanner . . . . .	12
2.4 Attenuation . . . . .	12
2.5 The SPECT scanner . . . . .	14
2.6 The PET scanner . . . . .	16
2.7 Intrinsic resolution in PET . . . . .	18
2.8 2D and 3D acquisitions . . . . .	19
2.9 References . . . . .	19
<b>3 Scanner models</b> . . . . .	<b>21</b>
3.1 Phantoms . . . . .	21
3.2 The Radon transform . . . . .	22
3.3 The discrete Radon transform . . . . .	23
3.4 Basic SPECT model . . . . .	24
3.5 Basic PET model . . . . .	27



3.6	Attenuation model . . . . .	29
3.7	General linear models . . . . .	29
3.8	Noise model . . . . .	30
3.9	Software . . . . .	32
3.10	References . . . . .	33
<b>4</b>	<b>Classical reconstruction . . . . .</b>	<b>35</b>
4.1	Methods and abbreviations . . . . .	35
4.2	Back-projection . . . . .	35
4.3	Examples of back-projection . . . . .	37
4.4	Filtered back-projection . . . . .	38
4.5	Examples of FBP . . . . .	41
4.6	EM reconstruction . . . . .	42
4.7	Examples of EM . . . . .	45
4.8	Other algorithms . . . . .	46
4.9	References . . . . .	47

## Part II: Improved Reconstruction

<b>5</b>	<b>Reconstruction goals and strategies . . . . .</b>	<b>51</b>
5.1	User requirements . . . . .	51
5.2	Technical requirements . . . . .	53
5.3	Statistical reconstruction . . . . .	55
5.4	Linear reconstruction . . . . .	57
5.5	Regularized reconstruction . . . . .	58
5.6	Incorporating anatomy . . . . .	61
5.7	References . . . . .	62
<b>6</b>	<b>Weighted least-squares . . . . .</b>	<b>65</b>
6.1	Chapter overview . . . . .	65
6.2	Likelihood functions . . . . .	66
6.3	Weighted least-squares . . . . .	67
6.4	Model-weighted least-squares . . . . .	67
6.5	General reconstruction . . . . .	68
6.6	Hausdorff distance . . . . .	69
6.7	Log-likelihood range . . . . .	71
6.8	Analysis of weighted least-squares . . . . .	72
6.9	Analysis of model-weighted least-squares . . . . .	73
6.10	Summary of the analysis . . . . .	74

## Part III: Numerical Algorithms

<b>7</b>	<b>Iterative framework</b>	<b>79</b>
7.1	Partial SVD	79
7.2	Diagonalizing the NPSVD	82
7.3	Updating the NPSVD	82
7.4	Updates with reorthogonalization	85
7.5	Krylov algorithms	86
7.6	Reorthogonalization	89
7.7	Restarting Lanczos	90
7.8	Restarting the NPSVD	92
7.9	References	93
<b>8</b>	<b>Computing the SVD</b>	<b>95</b>
8.1	On the meaning of “large-scale”	95
8.2	The memory hierarchy	96
8.3	Left residual factorization	97
8.4	Accuracy of matrix-vector product	98
8.5	Basic iteration	99
8.6	Accuracy and aliasing	100
8.7	Singular value decomposition	102
8.8	Explicit deflation	102
8.9	Implicit deflation	104
8.10	Sliding window	104
8.11	References	105
<b>9</b>	<b>Solving linear systems</b>	<b>107</b>
9.1	Prelude to iterative solvers	107
9.2	Bidiagonalizing solver	108
9.3	Unregularized solver	109
9.4	Regularized solver	110
9.5	Hybrid solver	111
9.6	References	112
<b>10</b>	<b>BBTools</b>	<b>113</b>
10.1	Introduction to BBTools	113
10.2	Central features	114
10.3	Working with black-box operators	117
10.4	Compatibility	120
10.5	References	123

## Part IV: Results

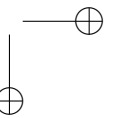
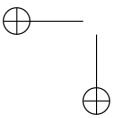
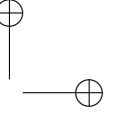
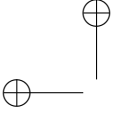
<b>11 Results</b> . . . . .	<b>127</b>
11.1 Introduction . . . . .	127
11.2 Low-count reconstruction . . . . .	127
11.3 Medium-count reconstruction . . . . .	132
11.4 Using anatomy in reconstruction . . . . .	134
<b>Conclusion</b> . . . . .	<b>136</b>

## Part V: Appendix

<b>A Addendum to WLS</b> . . . . .	<b>141</b>
A1 Image-space bound . . . . .	141
A2 Noise-perturbation of WLS-surface . . . . .	142
<b>B Base iteration</b> . . . . .	<b>143</b>
B1 Stopping criteria . . . . .	143
<b>C Matrix analysis</b> . . . . .	<b>147</b>
C1 Perturbation theory . . . . .	147
C2 Unitarily invariant norms . . . . .	147
C3 Norm inequalities . . . . .	149
C4 Matrix approximations . . . . .	149
C5 Near-orthogonal matrices . . . . .	150
C6 Triplet orthogonality . . . . .	151
C7 Accuracy of singular values . . . . .	153
C8 Triplet accuracy and aliasing . . . . .	155
C9 One-sided reorthogonalization . . . . .	155
<b>D BBTools reference</b> . . . . .	<b>157</b>
<b>Bibliography</b> . . . . .	<b>160</b>
<b>Index</b> . . . . .	<b>166</b>
<b>Nomenclature</b> . . . . .	<b>171</b>
<b>Papers<sup>1</sup></b> . . . . .	<b>Attached</b>

A cache-friendly kin to bidiagonalization with orthogonalization and restart (submitted)

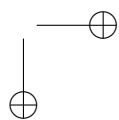
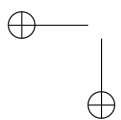
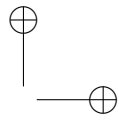
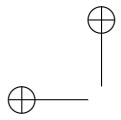
<sup>1</sup>The paper is excluded from the thesis, as the copyrights have been assigned to the publisher.





**Part I**

**Propaedeutic**



*This chapter outlines the contents of the thesis, and establishes basic concepts and typographic conventions.*

## 1.1 Tomographic reconstruction

In Greek  $\pi\rho\omicron$  (pro) means “before” and  $\pi\alpha\iota\delta\epsilon\upsilon\epsilon\iota\nu$  (paideuein) means “to teach”, so propaedeutic means a preparatory study. By using this word I hope to maximize the chance of getting the readers attention.

The material covered in this chapter is assumed knowledge throughout the thesis. As a motivation for reading the chapter, it explains how to identify chapters and sections you may skip without missing too much of the bigger picture.

The word  $\tau\acute{o}\mu\omicron\sigma$  (tomos) is Greek and means a slice or a cut and  $\gamma\rho\alpha\phi\hat{\omicron}$  (graphô) means a drawing. Thus tomography is the science of drawing slices. By making many cuts through an object, a 3D description of the target can be obtained.

Reconstruction

Of course the cuts needs to be retrieved without actually doing cutting, when we are dealing with medical imaging. Instead we make indirect measurements of some function of interest and tries to “reconstruct” the original function. What is recorded and how it needs to be reconstructed depends on the physical process involved.

## 1.2 Thesis outline

One fact can not be hidden and we may as well get it settled right away. This project was made by an engineer and the text will unavoidably have a certain perspective. However, the intended readership includes doctors and others without extensive formal mathematical training.

The problem is clear: parts of the thesis make heavy use of math to accomplish an objective which is (hopefully) interesting to users who will be unable to read it. I have done my best to encapsulate the mathematics in separate chapters, and make non-technical chapters explaining the rationale independent of these.

A disadvantage of this approach is that it sometimes breaks a natural flow, and occasionally requires forward references. I decided that overall this was preferable to the alternatives.

The thesis is divided into the following parts.

### **Propaedeutic**

Part I

The remaining chapters in this part describes the scanner physics (chapter 2), the data they produce (chapter 3), and the classical reconstruction algorithms (chapter 4).

## Improved reconstruction

Part II First we describe the constraints faced by an improved reconstruction method, and the two main reconstruction strategies (chapter 5). Based on the requirements, we select the method to be pursued, and settle on a strategy for including anatomy.

This is followed by chapter 6, which contains a technical argument for the competitiveness of the selected method compared with a large class of algorithms.

## Numerical algorithms

Part III This part is technical, and derives the algorithms that performs the actual reconstruction. It gives an iterative framework for Krylov algorithms (chapter 7), ultimately leading to the Numerical Partial SVD (NPSVD). It is shown how this can be used for large-scale SVD computations (chapter 8) and solving large-scale linear systems with and without regularization (chapter 9). Finally, we describe the toolbox BBTools (chapter 10), which contains efficient implementations of these algorithms.

## Results

Part IV Chapter 11 starts by showing that the new algorithm can reproduce the results from the Filtered Back-Projection algorithm. We then show that it may also be used to create images with higher resolution at the same noise suppression, and that any reconstruction in between these two extremes can be achieved in real-time by the operator.

Furthermore, we show that anatomical restrictions can be incorporated directly into the reconstruction. This can be accomplished in a way where the operator is fully in control of the process.

We follow up with a conclusion in chapter 11.4.

## 1.3 On the writing style

As recommended by John Kirkman [41], most of the thesis is written in an active first-person style, to the extent allowed by my ability on English. Most of the thesis is in plural, and singular is reserved mainly for personal observations, opinions, and decisions about the text itself.

In order to maintain a high level of readability, I have tried to maintain a consistent notation throughout the report, as defined in the nomenclature on p. 171. To further enhance the reader's focus on the contents rather than the text, I hope that the typographical quality provided by the typesetting package  $\text{\LaTeX}$  will keep such annoyances at a minimum.

## 1.4 Coordinates and directions

The coordinates of interest in this thesis are defined with respect to the scanner. Usually a patient rests on the back and is looking upwards, and the spine is thought of as the "human axis" and is defined as the z-axis, with the positive directions going from the feet to the head. The y-axis goes upwards and x-axis is positive in the left direction as seen from the patient.



Unfortunately, the coordinate system resulting from this definition is left-handed, going against deeply grown habits in the engineering/mathematical community. However, using a right-handed system would be just as confusing for the medical society, and thus the definitions above are adopted.

The terms left and right are always relative to the person in the scanner.

## Planes

There are several ways to define planes anatomically, and these may in turn be used to establish a coordinate system [79]. Some are defined with respect to cranial features and others are based on anatomical landmarks. For this thesis we do not need the precise definitions, and the meaning of the following terms can be given roughly:

- **Sagittal:**  $yz$ -plane. These are planes parallel to the *median* plane, which separates the two hemispheres of the brain. This corresponds to seeing the person from the side.
- **Transverse/transaxial:**  $xy$ -plane. The transverse planes are horizontal for a person standing up. A transaxial plane is like looking at a person from above.
- **Coronal:**  $xz$ -plane. Orthogonal to the two other planes and corresponds to looking at a person from behind.

In radiological practice the brain is often tilted in order to record the whole brain with a fewer number of slices. We will not distinguish between the differences in this thesis.

## 1.5 Linear algebra

∫∫∫

Sections, marked like this one, contains mathematical derivations. It is used in general chapters as a signal to people who prefer to skip the mathematical parts. Chapters full of mathematics can be identified from the introduction.

The thesis assumes a basic knowledge of linear algebra and numerical algorithms. Nonetheless, different sources use slightly different notation and definitions. Although most notation is introduced the first time it is used, the nomenclature given in the back, starting on page 171, should settle such ambiguities.

In any case, the most important rules are given here:

- Matrices are written in uppercase roman bold, e.g.  $\mathbf{A}$ .
- All vectors are columns. They are printed in lowercase roman bold, e.g.  $\mathbf{x}$ .
- We occasionally use the term “ $\mathbf{A}$  is orthogonal” as a short-hand for “ $\mathbf{A}$  has orthonormal columns”.
- The Hermitian of an operator,  $\mathbf{A}$ , is the Hermitian adjoint operator of  $\mathbf{A}$ . If  $\mathbf{A}$  is a matrix, the Hermitian is the transpose-conjugate of  $\mathbf{A}$ , i.e.  $\mathbf{A}^H$ . Thus, the Hermitian of an operator is not to the same as saying that  $\mathbf{A}$  is Hermitian or self-adjoint (which, for matrices, say that  $\mathbf{A} = \mathbf{A}^H$ ).

If a matrix is “small” it is with respect to the dimensions. Other interpretations will always be specified such as “ $\mathbf{A}$  is small in norm” or  $\|\mathbf{A}\|$  is small.

When a key-equation is derived or defined, it will either be encapsulated in a box or be given a name (or both). I hope the burden of reading is lightened when an equation is referred to by name rather than merely by a number such as eq. (7.12).

## 1.6 Software and programs

```
001010
100001
111100
```

The following quote is attributed to folklore in [14]:

The difference between theory and practice is smaller in theory than it is in practice.

Almost all the theory in the thesis has been fleshed out as working computer programs written in MATLAB<sup>®</sup>, which is a convenient language for numerical algorithms. This is particularly true in linear algebra which forms the back-bone of this thesis. They have been collected in a toolbox, BBTools [33], which is described in chapter 10, and is available under the GNU General Public License (GPL) as published by the Free Software Foundation [24].

Several examples in the text assume that the BBTools is installed. This allows much of the code to be expressed in a form where the algorithms are undistorted by the algorithmic details of the underlying linear function.

Sections mainly concerned with computer implementations are marked as this one. I personally prefer texts that put mathematical derivations into usage at the earliest possible opportunity, but I realize that this view is not universal.

Programming sections may include code or details that may appear inappropriate for those who are more mathematically oriented. Persons who feel that way should easily be able to skip these sections, possibly leaving them for a second reading.

## 1.7 Literate programming

```
001010
100001
111100
```

When it comes to algorithms this thesis differs from most texts. To put it in a nutshell: the algorithms in this thesis work. If the reader can not get an algorithm to work, it must be because the author made a bug. This is in contrast to the use of pseudo-code where it could be a misunderstanding from the reader. It should also be a comfort to know that the author actually ran each algorithm.

Although this approach makes the algorithms more verbose than pseudo-code, it does have the advantage of telling the truth, the whole truth and nothing but the truth. Most people who have tried to implement a program from pseudo-code should appreciate this quality.

Nevertheless, most source-code is sadly quite undocumented. Most examples in this thesis take their onset on literate programming, as described by its inventor Donald Ervin Knuth [42]:

Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a *computer* what to do, let us concentrate rather on explaining to *human beings* what we want a computer to do.

Unfortunately, the existing tools for literate programming are little but advanced pretty-printers. In particular, line-comments are not supported well. The value of good line-comments are evident from a quick glance at respected codes such as [58].

As a result, the code presented in the thesis was produced by a custom set of tools. The most important feature is that scopes has been replaced with connector-lines. The idea is that everything inside a scope shares a common line on the left, except for out-dented expressions. Apart from that, the most important feature is that any identifier can be replaced with an arbitrary command. Thus it is possible to match the notation in the program with the remaining parts of the thesis.

In MATLAB<sup>®</sup>-scripts (see alg. 1.1) every closing scope can be replaced by an **end** except for functions where nothing should be inserted. The branching construct shows the scoping rules. Since the whole **if**-line is an out-dented expression, it should be considered to be in the scope of the whole function. The same argument goes for the **else**-statement but not the assignment on the same line.

### Efficiency

Examples included in the thesis are meant to be illustrative, not efficient. Operations may be repeated, zero by construction, or never used. In particular, language-dependent optimizations are intentionally avoided, although they often have tremendous impact in speed. Examples include preallocating matrices and reformulations such as  $\mathbf{A}^H \mathbf{x} \rightarrow (\mathbf{x}^H \mathbf{A})^H$  and  $\mathbf{A}^{-1} \mathbf{x} \rightarrow \mathbf{A} \backslash \mathbf{x}$ .

For efficient code, please consult the code in BBTools [33].

## 1.8 Numerical notation

In order to analyze numerical algorithms, we need some notation. As most texts dealing with rounding errors in floating point arithmetics, we assume there is a constant,  $\epsilon_M$ , so that  $a \odot b = (a \circ b)(1 + \eta)$  where  $|\eta| \leq \epsilon_M$ . Here  $\circ$  is one of the basic arithmetic functions addition, subtraction, multiplication, division, or square root.

### Algorithm 1.1: Example of MATLAB<sup>®</sup>-script

*A nonsense demonstration showing how a MATLAB<sup>®</sup>-script is typeset in the thesis. The source is not shown in actual programs. The **end** that closes the **if**-structure is not shown since it collapses into an end-scope symbol.*

	Source	Demonstrates
<b>function</b> $a = \text{format\_demo}(\mathbf{A})$		
$\mathbf{x}_1 = \mathbf{A}(:, \text{end});$	<code>x1=A(:,end);</code>	Vector and matrix
$\beta = \ \mathbf{A}\ _2;$	<code>beta=norm(A);</code>	Scalar, matrix and standard function
<b>if</b> $\beta > \text{length}(\mathbf{x}_1)$	<code>if beta&gt;length(x1)</code>	Start of scope
$a = \text{length}(\mathbf{x}_1);$	<code>a=length(x1);</code>	Normal identifier
<b>else,</b> $a = \beta;$	<code>else, a=beta;</code>	Outdenting and multiple commands

The constant  $\epsilon_M$  is known as *machine precision* or *machine epsilon*. The relation is only valid as long as the result stays within the range of the arithmetic. While we may occasionally consider underflow (e.g. results flushed to zero or quantized to lower accuracy), it will generally be assumed that overflow does not occur.

For completeness it is usually mentioned that a few old computers (like the Cray C90 [15]) does not satisfy the relation for addition and subtraction. Such machines will be ignored, and we note that common software such as MATLAB<sup>®</sup> have abandoned such machines.

For linear algebra it is practical to have some bounds on a higher level. Specifically, we will consider the precision of an inner product of two vectors of length  $l$ . It can be proved (barring overflow and underflow) that accumulating the sum is stable, and computes the sum of the elements perturbed by a relative error smaller than  $\nu_l = l\epsilon_M/(1 - l\epsilon_M) \approx l\epsilon_M$ , where  $l$  is the number of entries added<sup>1</sup> [69].

---

<sup>1</sup>To account for the multiplication, we need to multiply this by  $\epsilon_M(1 + \epsilon_M)$  in the real case.

# Scanner instrumentation

Here the basic principles behind the scanners are introduced. We show example images from each scanner type, and explain the physics behind the CT, PET, and SPECT scanners. The MR scanner is only mentioned briefly, as the principles differ significantly from the others.

This chapter serves as a primer for those who want to understand the principles behind the scanners. It may also be beneficial for people who just want to know a bit about medical scanners in general.

## 2.1 Scanners and their abbreviations

There are 4 kinds of tomographic scanners used in medical applications that are capable of imaging any part of the body. They are all identified by abbreviations as given in table 2.1. Some synonyms found in the literature are given in table 2.2, but they will not be used further in this thesis. For completeness it should be mentioned that other methods exist, with ultra-sound probably being the most well-known.

**Table 2.1:**  
Acronyms for scanners and modalities used throughout the thesis.

Acronym	Meaning	Type
CT	Computer Tomography	Transmission
PET	Positron Emission Tomography	Emission
SPECT	Single Photon Emission Computerized Tomography	Emission
MR	Magnetic Resonance	Magnetic

The first scanner to emerge was the CT-scanner invented independently by Allan MacLeod Cormack and Sir Godfrey Newbold Hounsfield who shared the 1979 “Nobel Prize in Physiology or Medicine”. It is based on x-rays but instead of taking a single 2D photo, a large number of images are taken from different angles. These are then fed into a computer (hence the name) and the final images generated as explained in sec. 2.3. Since the x-rays are sent through the object, a CT-scan is known as a transmission scan.

This is in contrast to PET and SPECT, where a small dose of a radioactive substance is administered to the subject. Common examples are injections of water with a radioactive isotope of oxygen atom or inhalation of radioactive xenon.

**Table 2.2:**  
Other acronyms found in the literature. These will not be used in this thesis.

Acronym	Meaning	Explanation
CAT	Computer Aided Tomography	Same as CT
	<i>or</i> Computerized Axial Tomography	
ECAT	Emission CAT	Both PET & SPECT
MRI	Magnetic Resonance Imaging	Same as MR
NMR	Nuclear Magnetic Resonance	Same as MR
SPET	Single Photon Emission Tomography	Same as SPECT

When the radioactive atom decays a photon (in SPECT) or a positron (in PET) is expelled. In both cases the scanner tracks the direction of the decay, ultimately leading to a map of the concentration of the administered drug. Since both methods rely on emission from the substance itself, the methods are known as emission tomography.

Finally MR uses a completely different principle. An MR-scanner exploits that a nucleus in a magnetic field precesses around an axis. This precession can be manipulated with radio-waves, allowing different characteristics of the nucleus to be determined. In medical MR it is primarily protons (i.e. hydrogen) that are in play, and characteristics such as proton density can give very detailed anatomical information.

The importance of MR in this thesis is that its high resolution provides anatomical information that is useful for improving the resolution in PET and SPECT.

### Image gallery

The best way to get acquainted with the different scanner types is probably to see some images, as they usually come out of the scanners. Fig. 2.2 to 2.5 display sample images from all the scanners discussed above. This shows the relative strength and weaknesses of the scanners, and can safely be skipped by those who are already familiar with such images.

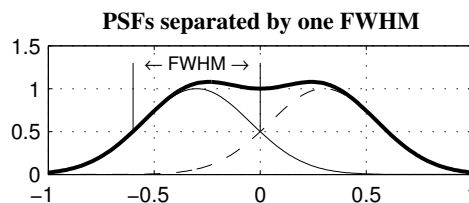
## 2.2 Resolution

Before discussing the scanners, we need a precise definition of resolution. As shown in the image gallery, the scanners produce images that are blurred versions of the function we would really like to measure.

In medicine, the resolution is usually quantified by measuring the Full-Width at Half Maximum (FWHM). This concept is associated with a Point Spread Function (PSF), which is the function measured of a point-source. Ideally, this would be a sharp spike, but in practice it is a function with a peak at the point-source that gradually tapers off.

The importance of the FWHM is illustrated in fig. 2.1. Since the PSF are, by definition,  $1/2$  of the peaks at the intersection point, the sum at this point is the average of the peaks. If the PSFs are monotonously decreasing, two peaks of equal intensity more than one FWHM apart will result in two crests separated by a cleft.

**Figure 2.1:**  
If the PSF decreases monotonously, then two equally intense peaks can be distinguished when they are separated by more than one FWHM.



This definition generalizes to any dimension. Generally, the FWHM is either measured as the longest distance or, if it is an ellipse, the major and minor axes are given separately.

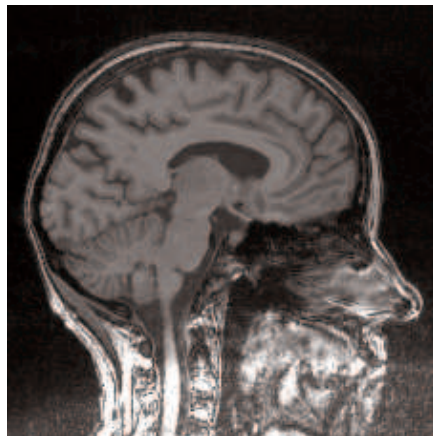


(a) Sagittal CT-image of a normal person.

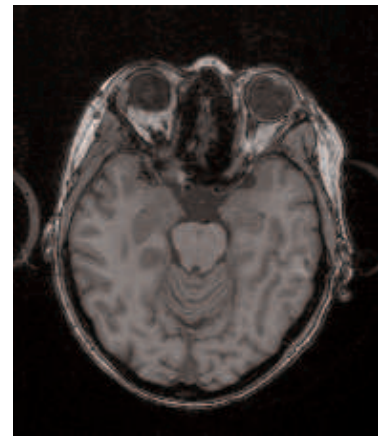


(b) Transaxial slice of the same person.

**Figure 2.2:**  
Example CT images. CT offer the best resolution of the ordinary scanners, but does not have a good contrast in brain tissue.

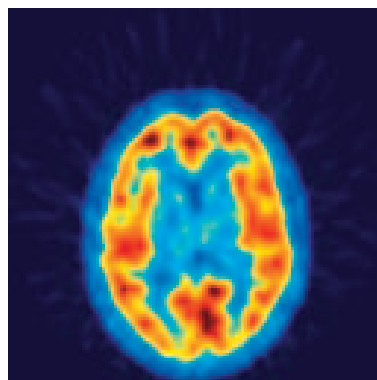


(a) Sagittal MR-image of a normal person.

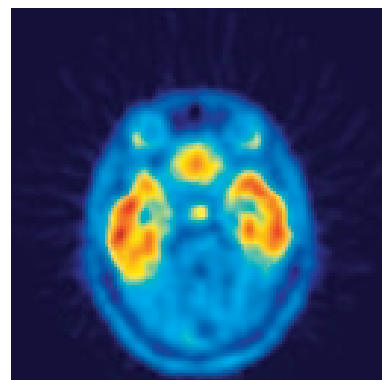


(b) Transaxial slice of the same person.

**Figure 2.3:**  
Example MR images. MR and CT are the primary means for identifying anatomical structures in the brain.

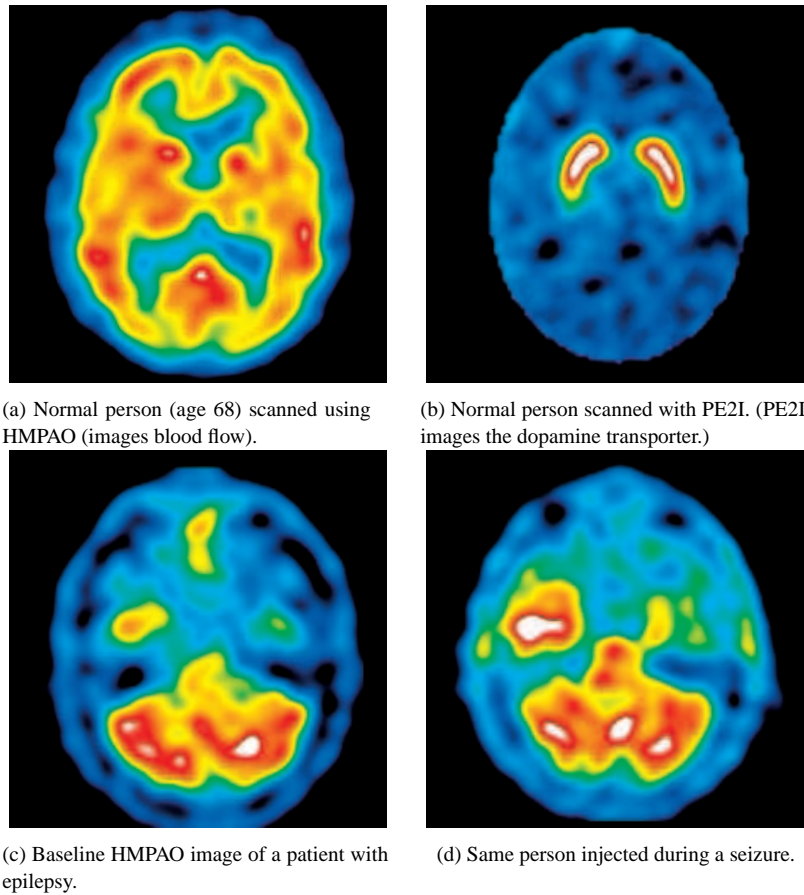


(a) Transaxial cut of an altanserin scanning. (Altanserin images a serotonin receptor.)



(b) Another cut, corresponding roughly to fig. 2.3(b)

**Figure 2.4:**  
Example of images produced with PET on the same person as in fig. 2.3. PET is a difficult technique, and requires a facility with a large staff of physicists and chemists in addition to the medical staff.



**Figure 2.5:**  
 Transaxial images produced using SPECT. The poor resolution limits the utility of the technique.

## 2.3 The CT scanner

Transmission tomography investigates a semi-transparent object by sending radiation through it from a number of angles as sketched in fig. 2.6. The projections are then fed to a computer and the amount of absorption is reconstructed. In medical imaging the relevant radiation is x-rays or  $\gamma$ -radiation. In the case of x-rays the scanning is called a CT-scan as an initialism (i.e. an unpronounceable abbreviation) for Computer Tomography. The term transmission scan is broader, and includes  $\gamma$ -radiation.

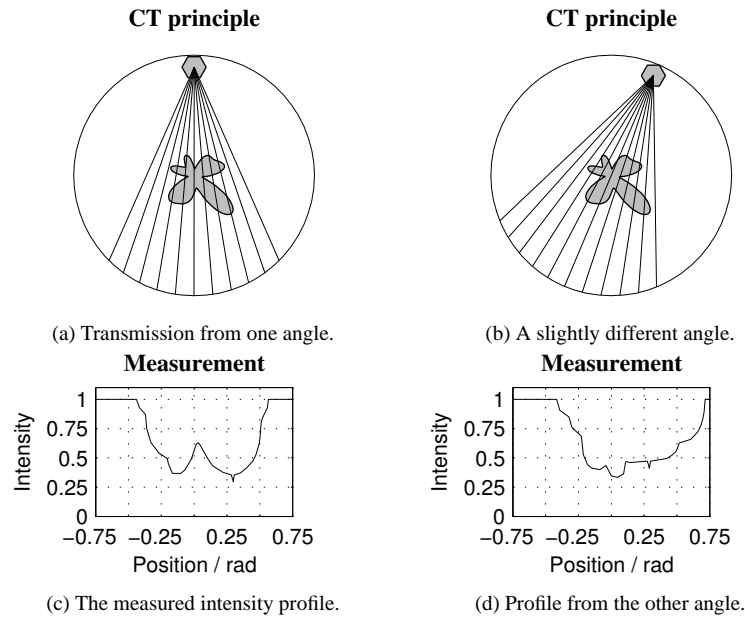
The CT-scanner is not only important in itself but also in emission tomography. First, emission tomography resembles transmission tomography from a technical point of view. Second, a transmission scan is often performed in addition to an emission scan in order to correct for attenuation of the radioactive tracer. Finally, the higher resolution of a CT-scan can be used to identify anatomical structures which can be used to interpret the emission scan or even improve it.

## 2.4 Attenuation

$\iiint$

As the x-ray traverses through an object, the ray is attenuated according to the tissue-type. The simple example in fig. 2.6 only has two types, but in general





**Figure 2.6:**  
 Principle of a modern CT-scanner. An x-ray source in the periphery of the scanner ring is rotated around the center. Each position of the source gives a projection as illustrated.

it can have many different values. Physics tells us that the intensity through a homogeneous absorbing substance can be written as follows:

$$(2.1) \quad f(x) = I_0 e^{-\mu x}$$

Here  $x$  is the length through the medium,  $\mu$  depends on the material, and  $I_0$  is the intensity of the incoming beam.

The coefficient  $\mu$  is known as the *linear attenuation coefficient*, and it depends on the substance as well as the energy of the photons (i.e. the wavelength) and has dimension of 1/length (1/m in SI-units). Most tables do not list  $\mu$  directly but rather the *mass attenuation coefficient*  $\mu/\rho$  where  $\rho$  is the density (usually given in  $\text{g}/\text{cm}^3$ ).

Some typical values for relevant energies are found in table 2.3. It can be seen that the main difference between the absorption is due to differences in the tissue density. In neurological studies bone is the only tissue with a density that differs significantly from water. Therefore CT is not usually the method of choice for retrieving anatomical information about brain tissue, except when a contrast agent<sup>1</sup> is used. However, the access to MR scans are limited and CT provides valuable information in many cases, and is an excellent technique for identifying intracranial bleeding.

When the attenuation is a function of position, the line must be split up in small segments. For simplicity we let the segments be distributed equidistantly with spacing  $\Delta x$ . This gives  $f(x) = I_0 e^{-\Delta x \mu(x_0)} e^{-\Delta x \mu(x_1)} \dots e^{-\Delta x \mu(x_N)}$  or simply  $f(x) = I_0 e^{-(\Delta x \mu(x_0) + \dots + \Delta x \mu(x_N))}$ . In the limit of an infinite number of segments this becomes  $f(x) = I_0 e^{-\int_0^x \mu(x) dx}$ .

This means that the measured profiles in fig. 2.6(c) and 2.6(d) contains information about the line-integrals of the attenuation-map since  $\int_0^x \mu(x) dx = \ln(-$

<sup>1</sup>A contrast agent is a substance with an attenuation that differs significantly from the area of interest. This property makes it visible in the CT scan.

**Table 2.3:**

Absorption in different tissues. The mass attenuation coefficient (MAC) and nominal densities (from [36]) are used to get the linear attenuation coefficient (LAC). The last column gives the absorption through one cm of tissue. Typical photon energies are 70–80 keV or lower for CT, 100–150 keV for SPECT, and 511 keV for PET.

Tissue	Density ( $\rho$ ) g/cm <sup>3</sup>	Photon keV	MAC ( $\mu/\rho$ ) cm <sup>2</sup> /g	LAC ( $\mu$ ) cm <sup>-1</sup>	Absorption % pr. cm
Bone	1.92	500	0.090	0.173	16
		100	0.186	0.356	30
		50	0.424	0.814	56
Water	1.00	500	0.097	0.097	9
		100	0.171	0.171	16
		50	0.227	0.227	20
Blood	1.06	500	0.096	0.102	10
		100	0.170	0.180	16
		50	0.228	0.241	21
Brain <sup>†</sup>	1.04	500	0.096	0.100	10
		100	0.170	0.177	16
		50	0.228	0.237	21

<sup>†</sup> Brain includes both gray- and white-matter as they are virtually indistinguishable.

$f(x)/I_0$ ). In the general case,  $x$  represents the position on the line inside the volume.

In fig. 2.6 each point on the graph represents such line integral. In this simple example the  $\mu$ -function is either zero or a constant, say,  $c$ . Thus the logarithm of the measurement will be  $-c$  times the length that the line intersects the object.

The  $\mu$ -function is sometimes called a  $\mu$ -map. The CT-images are slices computed from these projections, usually trans-axial e.g. planes parallel to the floor when standing. The procedure for doing this will be explained in chapter 4.

## 2.5 The SPECT scanner

The SPECT-scanner is an emission tomographic scanner, i.e. the scanned object must itself be radiating. This allows the scanner to track a radioactive substance within the patient.

The principle of a basic SPECT-scanner is depicted in fig. 2.7. Much as the CT-scanner, the SPECT-scanner measures the radiation from a number of directions. However, since the object is radiating in all directions simultaneously, the  $\gamma$ -rays must be limited prior to the measurements. Also, most SPECT-scanners are multi-head scanners, i.e. there are two or more  $\gamma$ -cameras rotating simultaneously.

For completeness, we list a couple of tracers used in SPECT-studies in tab. 2.4. One noteworthy observation is that the half-lives of the tracers are sufficiently long that they can generally be created and transported from another site, e.g. a commercial vendor.

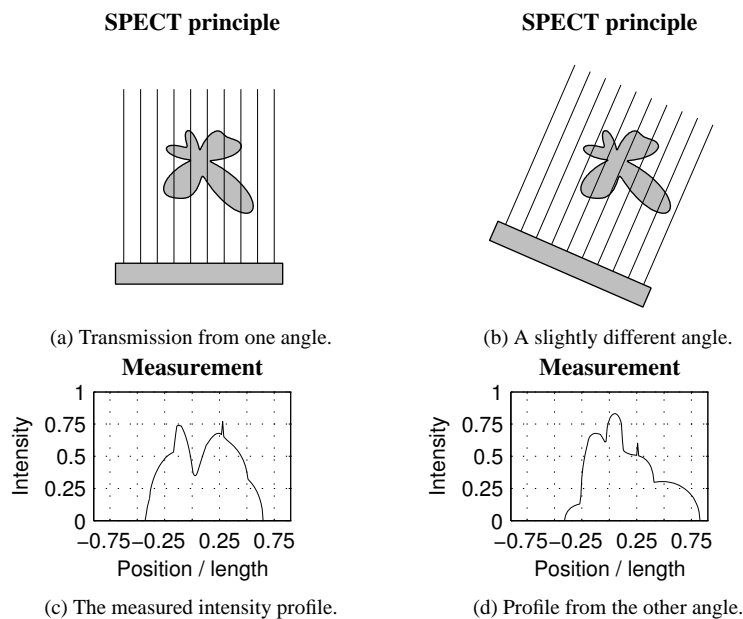
Most tracers used clinically are based on technetium (<sup>99</sup>Tc), which is produced by decaying molybdenum (<sup>99</sup>Mo). The half-life of <sup>99</sup>Mo is about 66 h, and can typically produce <sup>99</sup>Tc for about a week. The tracer is labeled with the technetium using a standard kit, making SPECT relatively inexpensive.

### Collimators

To get directional information from the beams, they are limited by a device known as a collimator. A collimator consists of a number of pipes (usually of lead), and

**Figure 2.7:**

Principle of a basic SPECT-scanner with a single head and a parallel-hole collimator. The activity of a radiating object is measured by a  $\gamma$ -camera. A collimator, placed in front of the camera, limits the radiation to beams normal to the surface. The camera is rotated around to object to produce measurements from a number of directions.

**Table 2.4:**

Examples of isotopes and tracers used in SPECT-studies.

Isotope	Half-life	Tracer	Use
$^{99}\text{Tc}$	6.0 h	HMPAO <sup>†</sup>	Blood flow
$^{133}\text{Xe}$	5.2 d	Xenon	Blood flow
$^{123}\text{I}$	13.2 h	PE2I	Receptor studies
		ADAM	Receptor studies

<sup>†</sup> HMPAO (Hexa-Methyl-Propylene Amine Oxime) is a lipophilic tracer used to diagnose e.g. strokes and dementia. PE2I, a cocaine derivative, is used to study dopamine transporters, while ADAM is used to study serotonin transporters.

a beam is only allowed to pass through when it is aligned with the tube. The scanner depicted in fig. 2.7 is a parallel-hole collimator, but many other geometries are possible.

A fan-beam collimator has all tubes directed towards a line parallel to the scanner z-axis. Thus, all trans-axial cuts through the collimator will have the same structure. A cone-beam collimator has tubes all directed at a single point in space. Finally, a pin-hole collimator consists of a single hole, much like a pin-hole camera.

The collimator acts as a filter and blocks the majority of  $\gamma$ -rays. Therefore, the collimator chosen for a task must be a trade-off between sensitivity (how many  $\gamma$ -rays are blocked) and resolution (how closely the beam is aligned with the pipes). Typical examples of the pipe dimensions of low-energy high-resolution and ultra-high-resolution collimators are given in tab. 2.5.

**Table 2.5:**

Examples of commercial collimators for the Philips (formerly Marconi) IRIX-scanner. (Private communications.)

Name	Geometry	Hole size mm	Hole length mm	Thickness mm	Focal length mm
LEHR-PAR	Parallel	1.22	27.0	0.20	$\infty$
LEUHR-PAR	Parallel	1.78	58.4	0.15	$\infty$
LEHR-FAN	Fan-beam	1.40	25.4	0.18	500
LEUHR-FAN	Fan-beam	1.40	34.9	0.15	500

Although SPECT does not have a theoretical lower resolution limit, the collimator introduces severe practical limits. First, to obtain a sensitivity which is useful in medical applications, it is necessary to allow beams that diverge from the ideal. Also, the thickness of the tubes are limited, allowing some photons to be reflected or penetrate the tube. In practice, the resolution of a SPECT-scanner is limited to about 8 mm in typical clinical studies.

These effects are minimal for a pin-hole collimator. However, to compensate for the poor sensitivity of such a collimator, it is usually necessary to inject a very high dose of radioactivity for a complete brain-scan. Normally, the high dose of radioactivity will kill the subject, which limits the utility to animal studies. The resolution, however, is excellent.

### SPECT imperfections

There are a number of imperfections that limits the resolution of SPECT. First, the attenuation discussed in sec. 2.4 is equally valid for SPECT. However, where the CT-scanner is used to measure the attenuation, it is a nuisance in SPECT we would rather have been without.

Typically, the energies in SPECT are higher than in CT. As can be seen in tab. 2.3, the attenuation becomes less problematic for the higher photon energies. Furthermore, the difference between the mass attenuation coefficient for bone and water decreases rapidly in the energy-range 100–120 keV.

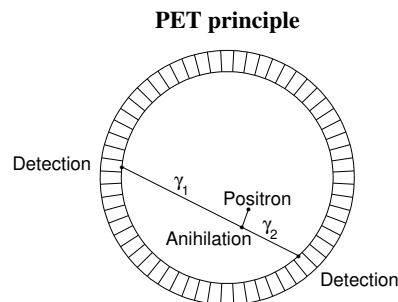
Secondly, some of the “attenuated” photons are not absorbed, but scattered in new directions. This comes at a loss of energy, which depends on the deflection angle. This effect can, to some extent, be disregarded by only considering photons in a certain energy window.

There are many other imperfections in SPECT, but it is beyond the scope of this thesis to explain them in detail. We refer the interested reader to the references in sec. 2.9.

## 2.6 The PET scanner

Like SPECT, PET is an emission tomographic technique. However, the PET-scanner is based on radioactive isotopes that has a  $\beta^+$  decay, i.e. emits a positron. Shortly after emission, the positron annihilates with an electron and they are both converted into pure energy in the form of  $\gamma$ -radiation. As it is impossible to conserve both energy and momentum with a single photon, the energy is converted into two  $\gamma$ -quanta of (nearly) opposite direction as sketched in fig. 2.8.

**Figure 2.8:**  
The basic principle of a PET-scanner. The ligand emits a positron, which quickly annihilates into two 511 keV  $\gamma$ -photons of opposite directions.



The PET-scanner consists of rings of scintillation crystals, and the line of response (LOR) of the annihilation is determined by the pair of crystals that detects the emission nearly simultaneously. This eliminates the need for a collimator, which makes PET far more sensitive than SPECT.

Because of the  $\beta^+$ -decay, most isotopes have a relatively short life as shown in table 2.6. This is both an advantage and a problem. It is a problem because the isotopes have to be produced at the place they are used. It is difficult to build the isotope into the tracer, test the purity and administer the drug during the short time available. One advantage is that the substance is quickly removed which reduces the dose absorbed by the subject and allows consecutive scans.

Isotope	Half-life	Substance	Use
$^{15}\text{O}$	2 m	Oxygen	Oxygen metabolism
		Carbon monoxide	Blood volume
		Water	Blood flow
$^{11}\text{C}$	20 m	Flumazenil	Receptor studies
		Raclopride	Receptor studies
$^{18}\text{F}$	110 m	FDG <sup>†</sup>	Glucose metabolism
		Altanserin	Receptor studies
$^{13}\text{N}$	10 m	Ammonia	Cardiac blood flow

**Table 2.6:**  
Isotopes and example of tracers used in PET-studies.

<sup>†</sup> FDG (Fluoro-Deoxy-Glucose) is a sugar used for activation studies and cancer treatment (i.e. oncology).

## PET imperfections

As in SPECT, there are a large number of non-ideal effects that ultimately limit the resolution in PET. It is beyond the scope of this thesis to go into details with most of them, but we briefly discuss a few with practical consequences.

### Physical limitations

There are several unavoidable physical effects that limit the resolution of a PET-scanner. First, the positron usually contains a residual momentum when it annihilates with the electron. Since momentum is conserved, the result is that the two beams are not completely collinear. This effect depends on the speed of the emitted positron, which depends on the tracer used in the study.

Secondly, there is a distance from the emission of the positron to the annihilation. This is called the “positron range”, and depends on the speed of the emitted positron and the nature of the surrounding tissue. This is further discussed in sec. 2.7.

Thirdly, attenuation and scatter have much the same effect on PET as it has on SPECT. However, if one of the photons is scattered, the measured line may not be inside the object. This makes scatter more severe for PET than it is for SPECT. As in SPECT, most (but not all) of the scattered photons can be identified by measuring the energy-loss by the photon.

### Equipment limitations

The scintillation crystals in the PET-scanner convert the 511 keV  $\gamma$ -rays to lower energies which can be detected using conventional electronics. Each crystal is

used to detect emissions that occurs on all the lines that intersects it. This is unlike SPECT where the crystal residing under a collimator tube only detects photons from (approximately) one direction.

This greatly increase the sensitivity of PET compared to SPECT. However, a crystal has a dead-time after measuring a photon where it returns to equilibrium and is unable to detect new events. This is a physical effect, and is not merely limited by the electronic circuits. The result is that the number of detected events is a non-linear function of the actual number of photons emitted on a line.

Most algorithms used to form images depends on the linearity of the response. The dead-time makes it necessary to correct the measurements before further processing. Fortunately, this is standard on modern PET-scanners.

### Randoms

Finally, we mention randoms which may be measured if two emissions occur at nearly the same time. If two of the four photons are absorbed, the measured LOR may be completely unrelated to the object.

The standard way to correct for randoms is to measure and subtract known randoms. These can be identified when two emissions are detected with a time-interval that is too large to be emitted from a point inside the scanner. Thus, there is a window of “acceptance” where counts are included, followed by a window of “randoms” of the same duration. When a random is detected, it is subtracted instead of added. This technique is known as on-line subtraction.

This arrangement is special because it allows “negative counts”. On the average, the measured number will be correct, but the existence of negative counts have implications for some algorithms. In principle, the “trues” and “randoms” can be measured independently, but not all PET-scanners support this.

## 2.7 Intrinsic resolution in PET

∫∫∫

Unlike SPECT, the positron range and non-collinearity puts a fundamental limit to the resolution of PET. The size of the scintillation crystals and other non-ideal properties of the PET-scanner further limits the resolution.

The positron range depends on the isotope, with some examples given in table 2.7. Here  $r$  is the FWHM-equivalent of the standard deviation and can thus be combined with other factors.

Isotope	$^{18}\text{F}$	$^{11}\text{C}$	$^{68}\text{Ga}$
FWHM [mm]	0.13	0.13	0.31
FW(0.1)M [mm]	0.38	0.39	1.60
$r$ [mm]	0.54	0.92	2.80

**Table 2.7:**  
Isotopes and typical ranges [16].

The effect arising from non-collinearity depends on the diameter of the detector ring,  $D$ , and at the center it is given roughly by  $.0022D$ . Even if technology could allow perfect detection, the intrinsic resolution of a PET-scanner would still be limited to  $R = 1.25\sqrt{r^2 + (.0022D)^2}$  [16]. For a dedicated brain-scanner this could be less than 1 mm for  $^{18}\text{F}$ -images.

Practical scanners must have crystals of finite width,  $d$ , and there may be a significant coupling between crystals which also limits the resolution. In this case

the range is approximately given by  $R = 1.25\sqrt{(d/2)^2 + r^2 + (.0022D)^2 + b^2}$ , where  $d$  is the crystal width and  $b$  is the coupling. Tomographs with a photomultiplier tube (PMT) for each crystal typical have  $b < .3$  mm, but for economic reasons crystals<sup>2</sup> often share PMTs. This usually limits coupling to  $b \approx 2$  mm.

Thus the achievable resolution of modern scanners useful for brain-scanning is around 4–6 mm, and 2 mm is a reasonable estimate of the best resolution that can be achieved with current technology.

## 2.8 2D and 3D acquisitions

So far we only discussed how a transaxial slice is acquired. In order to produce a 3D volume, a number of slices are measured simultaneously. In SPECT, the  $\gamma$ -camera acquires a number of plane-projections, and a line from each plane is used as the data for a transaxial slice. This may only be approximately true if the object moves during the acquisition, but this is usually ignored.

Modern PET-scanners can be operated in 2D and 3D mode. The PET-scanner consists of a number of rings, and in 2D-mode each ring collects the data for a slice. The number of slices can be nearly doubled by treating adjacent rings as a single ring, but we shall not distinguish between a “virtual ring” formed by a ring-pair and a real pair.

Although the object radiates in all directions, only beams that are not perpendicular to the scanner  $z$ -axis are desirable in 2D-mode. Much like the SPECT-scanner, other beams are limited using a metal shield. In PET, the shields that blocks this radiation are known as septa. The most common material for the septa is tungsten.

In 3D-mode the septa are removed during the acquisition. This drastically improves the sensitivity of the scanner, but also exacerbates the imperfections discussed in sec. 2.6.1. More importantly to the work in this thesis, the amount of data acquired during a scan increases enormously.

While the software developed in this thesis was intended to allow full 3D reconstruction, the memory requirements is at least 4 Gb for a practical problem. This can not be accommodated by the 32-bit computers available at NRU during the project. We shall therefore, with some regret, disregard 3D reconstructions from the thesis. We look forward to working on 3D-problems as 64-bit computers become mainstream.

## 2.9 References

Most of the material in this chapter is standard. The tome by Bushberg et. al. [12] includes almost everything described in this chapter, and can be recommended for both practioners and technical oriented people. It is particularly strong on CT-scanners, but also includes much material on gamma-cameras.

For a hands-on guide to SPECT, the primer by English [19] is a good starting point. However, it mostly leaves out technical details.

A good source for PET, for those who are technological oriented is [77]. Most of the descriptions of PET-instrumentation in this chapter was derived from this

<sup>2</sup>Actually slits of different depth are carved into a single crystal block.

book. A shorter, but informational, paper for PET is [74], which should be readable for most involved with PET.

Positron range is treated in [16], while analytical approximations for several isotopes can be found in [28].

For 3D-acquisition, algorithms, and scanner limitations, [5] is a thorough and complete reference. It is, however, primarily oriented towards people with a strong mathematical background.

Many issues regarding imperfect hardware in PET is treated statistically in [84]. Randoms, in particular, is treated this way in [85], while a hardware-oriented and experimental treatment can be found in [35].



# Scanner models

*This chapter develops mathematical models of the emission tomographic scanners. We define test-images, phantoms, which will be used throughout the thesis. We also show typical responses measured by the scanners in the form of sinograms.*

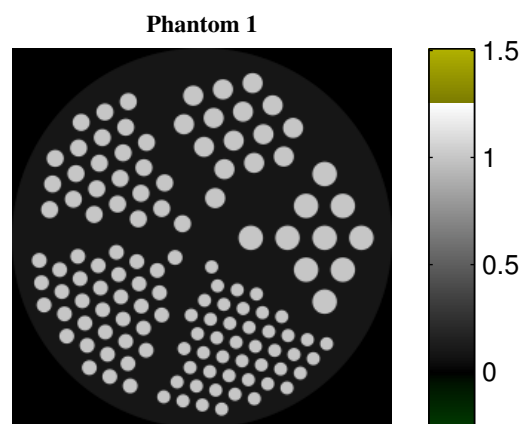
*The bulk of the chapter is mathematically oriented. We define the Radon transform, which is followed by specific models of the SPECT- and PET-scanner, followed by a more general linear model. We also describe the Poisson distribution, which is the fundamental statistical model underlying emission tomographic measurements.*

## 3.1 Phantoms

As a prelude modeling the scanners, we shall define some test-images which will be used to illustrate the procedures. In tomography, a test-image is called a phantom. Physical phantoms are usually made of plastic, and may contain a number of chambers. These chambers may in turn be filled with radioactive liquid, which can be measured separately. This allows a quantitative measurement, which can be used to evaluate the accuracy of a real reconstruction.

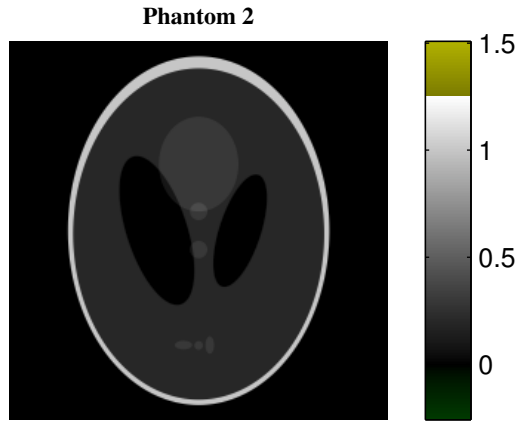
We shall use the following synthetic phantoms in this thesis. In fig. 3.1 we have a test-phantom, which is suitable for estimating the resolution. It simulates a number of rods, arranged in a hexagonal grid. The smallest rods which can be distinguished individually represents the resolution of the scanner.

**Figure 3.1:**  
*This phantom is useful in testing the resolution of a reconstruction image.*



The second phantom, depicted in fig. 3.2, is a classical phantom used in neurological studies. It consists of a sum of ellipses, which can be treated analytically. Although we shall not use this property, we use it because it is familiar to many. We shall use a variant due to Peter Aundal Toft [72], which has a better contrast than the original.

**Figure 3.2:**  
The Shepp-Logan phantom is a classical phantom used to in neurological tomography [72].



All phantoms and reconstructed images are represented as  $256 \times 256$  pixels. There is nothing magic about this size, except that images are traditionally rendered as a  $2^n \times 2^n$  for some  $n$ . For the purpose of this thesis, a size such as  $257 \times 257$  (i.e. a prime) would serve just as well.

For precision, we shall adopt the convention that all images represents a function,  $f(x, y)$ , defined in the range  $(x, y) \in [-1; 1] \times [-1; 1]$ , where  $\times$  is the cartesian product. By definition we set  $f(x, y) = 0$  for any value outside this area, i.e. it has *compact support*.

### 3.2 The Radon transform

∫∫∫

The scanners described in chapter 2, i.e. CT, SPECT, and PET, all measure the response along a large number of Lines Of Response (LORs). In the case of emission tomography, the measurements are the sum of events occurring on each LOR during the scan.

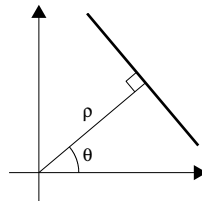
The number of events is proportional to the activity of the substance at each point. Thus, if  $\ell$  represent a LOR and  $f(x, y)$  is the activity at the point  $(x, y)$ , the measured activity is proportional to the quantity:

$$(3.1) \quad g(\ell) = \int_{\ell} f(x, y) dx$$

normal form

For this to be useful, we need to parameterize the line  $\ell$ . While several possibilities exist, the most convenient representation in tomography is the *normal form*, which is specified by the distance from origo,  $\rho$ , and the angle,  $\theta$ , as shown in fig. 3.3. It is worth noting that for a parallel-hole collimator in SPECT, the angle of the LOR,  $\theta$ , is the inclination of the collimator.

Normal form of line



**Figure 3.3:**  
A line represented in normal form.

Using this convention, the function  $g(\theta, \rho)$  is now a function of  $f(x, y)$  as specified by eq. (3.1). This is known as the Radon-transform, after Johann Karl August Radon [81] who showed that the functions were linked by a differential equation that has a unique solution under mild restrictions.

It follows directly from eq. (3.1) that the Radon transform is symmetric, i.e. it satisfies  $g(\theta, \rho) = g(\theta + k360^\circ, \rho)$ ,  $k \in \mathbb{Z}$ , and  $g(\theta, \rho) = g(\theta + 180^\circ, -\rho)$ . This means we only need to consider the interval  $0 \leq \theta < 180^\circ$ , or alternatively,  $0 \leq \theta < 360^\circ$  and  $\rho \geq 0$ .

The first form is by far the most convenient in tomographic problems, so we shall adopt this convention throughout. Furthermore, we only consider images with compact support. Specifically we have  $f(x, y) = 0$  for  $\max\{|x|, |y|\} > 1$ . Therefore, we have  $g(\theta, \rho) = 0$  for  $\rho > \sqrt{2}$ .

However, this range is often limited further by a field of view (FOV). In this case we define  $f(x, y) = 0$  for  $\sqrt{x^2 + y^2} > R$ , where  $0 < R \leq \sqrt{2}$  is the largest distance from the origin. In this case we have  $g(\theta, \rho) = 0$  for  $|\rho| > R$ .

### 3.3 The discrete Radon transform

Sinogram

The Radon-transform of an image is called a *sinogram*. This term stems from the fact that the Radon transform of a single point is a sine-function.

In practice, the Radon transform is discretized. First off, only a finite number of projections are made. Considering SPECT as an example, the simplest mode of operation is “step-and-shoot” where the heads moves to a location (a single value of  $\theta$ ), and remains there for some time. The angles are discretized to the positions of the head (or heads for multi-head scanners).

Even in “continuous” mode, where the heads moves slowly along a prescribed orbit, the data is collected in planes at discrete values of  $\theta$ . Here, however, each value represents an integral over the range of angles which result in a slight blurring of the images.

Secondly, the image is discretized. As described in sec. 3.1, the images in this thesis will always be  $256 \times 256$  pixels.

Finally, the position (i.e.  $\rho$ ) is discretized in a finite number of bins. When a scanner detects an event, it determines a bin corresponding to the LOR, and increases it. The number of bins depends on the scanner type and will be discussed below. For synthetic experiments, it is reasonable to sample the sinogram at the same density as the image. In our case, this would mean  $\rho$  is sampled at  $256\sqrt{2} \approx 362$  locations.

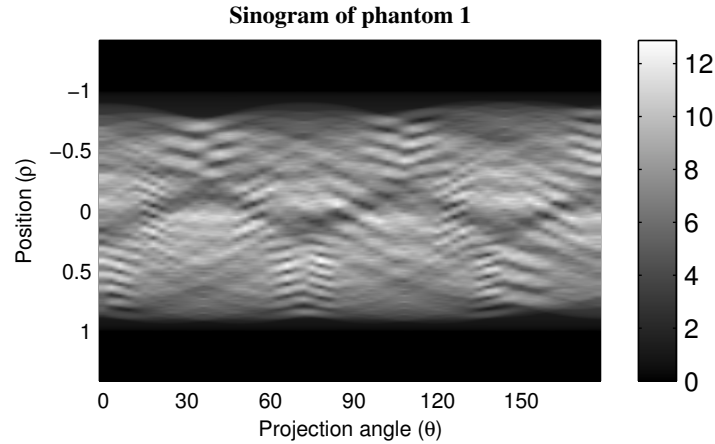
With these caveats, we can now show the sinograms of the phantoms in fig. 3.4 and 3.5.

#### Implementation issues

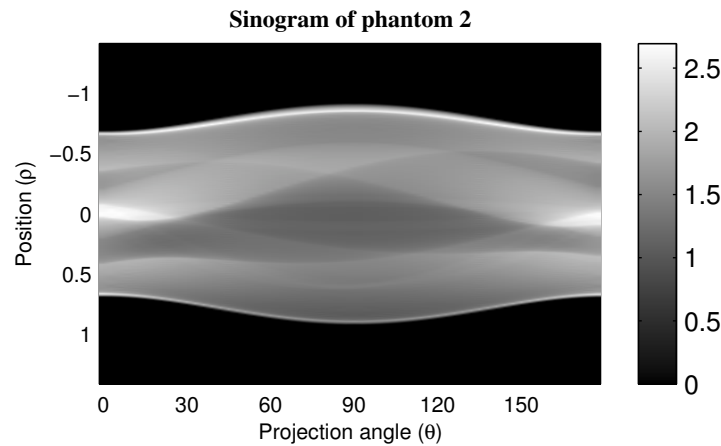
The first issue that arise concerning an implementation of the Radon transform is to decide how the image should be represented. There are several options, but a simple approach that carries over to more complex models, is to represent the image as a set of point-masses centered at pixels of equal size.

001010  
100001  
111100

**Figure 3.4:**  
The Radon-transform of the resolution-phantom in fig. 3.1, sampled every  $3^\circ$ . The sinogram is scaled so the bins sum to 100,000, which will be used to illustrate noise.



**Figure 3.5:**  
The Radon-transform of the Shepp-Logan phantom in fig. 3.2 with bins scaled to a sum of 20,000. The sinogram appears smoother than fig. 3.4 as the image has fewer sharp edges.



A problem with this approach is that the sampling density of  $\rho$  may conflict with the density of the point-masses for certain angles. This problem can be nearly eliminated by interpolating the sinogram using a squared-cosine, i.e. if a point of mass  $w$  is projected at sample 3.25 for some angle  $\theta$ , the mass is distributed to sample 3 and 4 as  $w \cos(.25\pi/2)^2$  and  $w \cos(.75\pi/2)^2$ .

Another option is to ensure the image is sampled densely compared to  $\rho$ .

### 3.4 Basic SPECT model

∫∫∫

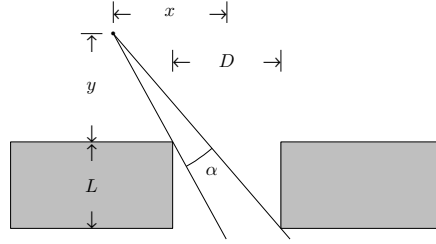
A collimator is a complex device, and we shall use a simple idealized model. Consider a single point emitting photons and a collimator of lead with a single cylindrical hole. In the ideal case, photons will only pass the collimator if they pass through empty space.

Our goal is to derive the fraction of the photons that pass the collimator as a function of position. Since the geometry is symmetric, we only need to consider the distance from the collimator,  $y$ , and the distance from the center of the hole,  $x$ . We use  $L$  to denote length of the hole, and  $D$  for the diameter. The geometry is shown in fig. 3.6.

### Single-hole collimator

**Figure 3.6:**

A collimator with a single hole with diameter  $D$  and length  $L$ . The figure is the plane of the hole axis and emission point.



An unshielded point-source radiates with equal intensity in all directions. The intensity of the beam is therefore proportional to the angle allowed to pass the collimator. We shall simplify the analysis by assuming that  $y \gg D + L$ , i.e. the hole has infinitesimal dimensions. This implies that  $\alpha$  is small, and the intensity is negligible. We shall therefore normalize the intensity to 1 for a source that is placed on the hole axis, i.e.  $x = 0$ .

With these conventions in mind, we first note the following function points:

$$(3.2) \quad I(x, y) = \begin{cases} 1 & \text{for } |x| \leq D/2, \\ 0 & \text{for } |x| \geq D/2 + y(D/L). \end{cases}$$

For intermediate points, the desired intensity is the ratio  $\alpha/\alpha_0$ , where  $\alpha_0$  is the angle for a point at the center:

$$I(x, y) = \alpha\alpha_0^{-1} = \left( \tan^{-1} \left( \frac{|x| + D/2}{y + L} \right) - \tan^{-1} \left( \frac{|x| - D/2}{y} \right) \right) \alpha_0^{-1}$$

Using the identity  $\tan^{-1}(x) - \tan^{-1}(y) = \tan^{-1}((x - y)/(1 + xy))$  [86]:

$$I(x, y) = \tan^{-1} \left( \left[ \frac{|x| + D/2}{y + L} - \frac{|x| - D/2}{y} \right] / \left[ 1 + \left( \frac{|x| + D/2}{y + L} \right) \left( \frac{|x| - D/2}{y} \right) \right] \right) \alpha_0^{-1}$$

Simplifying and ignoring second order terms:

$$I(x, y) \approx \tan^{-1} \left( \frac{Dy - L|x|}{x^2 + y^2} \right) \alpha_0^{-1}$$

Since the angle  $\alpha$  is infinitesimal, we may further use the relation  $\tan^{-1}(x) \approx x$ . The normalizing term,  $\alpha_0$ , is defined by the relation  $I(0, y) = 1$ , giving the pleasing result:

$$I(x, y) = \frac{1 - (L/D)(|x|/y)}{1 + (x/y)^2}$$

It is reasonable to simplify this even further. A practical collimator must limit the photons to small angles, which implies that  $L/D$  is much larger than one (see e.g. tab. 2.5 on p. 15). Therefore, the term  $x/y$  does not change appreciably from 0 in the valid range, i.e.  $|x| \leq y(D/L)$ .

We may combine this approximation with eq. (3.2), which simplifies under the assumption that  $L$  and  $D$  are infinitesimal, whereas the ratio  $L/D$  is non-negligible:

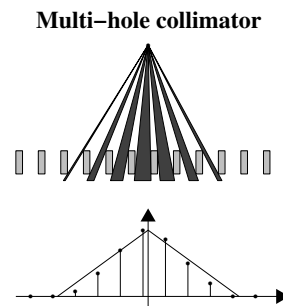
$$(3.3) \quad I(x, y) = \begin{cases} 1 - (L/D)(|x|/y) & \text{for } |x| \leq y(D/L), \\ 0 & \text{otherwise.} \end{cases}$$

### Multi-hole collimator

A single-hole collimator is not very useful, except for pin-hole collimators which require an entirely different analysis. However, when the response of a single hole is known, it is easy to determine the response of a collimator consisting almost entirely of holes.

Consider fig. 3.7, which shows the response of a number of holes. The measured response from a single point consists of the sum of the response of all the holes. The figure shows both the discrete values of the angles and the continuous model given by eq. (3.3). Clearly, the continuous approximation is quite good, even though the infinitesimal requirement is violated to make the drawing legible.

**Figure 3.7:**  
A multi-hole collimator. The continuous model, from eq. (3.3), is close to the ratio of angles, which is shown as discrete points.



We still need to take normalization into account. Real collimators will collect nearly the same number of counts regardless of the distance of the point. Therefore, the area under the curve is nearly constant, and the intensity under the point should be scaled accordingly.

If depth-dependent sensitivity needs to be modeled, it is probably best to measure this dependence directly from the scanner. Note that only the collimator response should be included in this. We shall treat attenuation separately.

### Model flaws

It should be noted that eq. (3.3) was derived from 2D-considerations only. A more detailed analysis involves the solid angle, and scaling becomes a somewhat complicated matter. Nevertheless, the result does not change appreciably when this is taken into account, and the assumptions used to derive the simple model is too far from reality to justify such an analysis.

In reality, collimator holes are not cylindrical as this would waste area. Neither are the holes infinitesimal, photons are reflected, scattered, and may penetrate the walls between the holes. The accuracy of the  $\gamma$ -camera is another limiting factor. In short, an exhaustive list of significant issues neglected by the model would be very long.

When these factors are combined, the measured point-spread function is much closer to a Gaussian function than the simple triangular function in eq. (3.3). Nevertheless, it captures an essential property of the collimator: beam divergence.

An important consequence of beam-divergence is that the resolution of a point decreases with the distance from the collimator. Therefore it is critical to get the

head of the scanner as close to the object as possible. It also means that the measurements can not adequately be represented by the limiting range  $0 \leq \theta < 180^\circ$ , but must be extended to the whole range  $0 \leq \theta < 360^\circ$ .

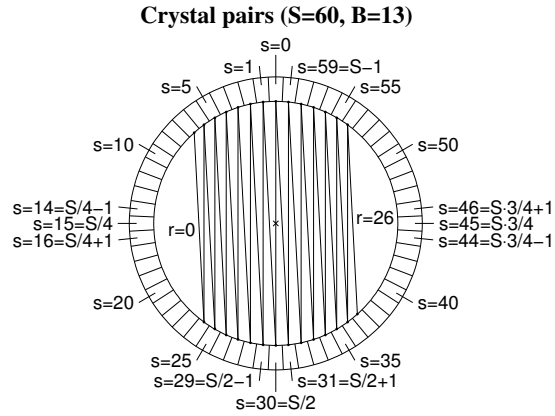
### 3.5 Basic PET model

∫∫∫

Fortunately, it is somewhat easier to model PET than SPECT, at least in 2D-mode. For simplicity, we shall consider a scanner with only  $S = 60$  crystals, but the observations carries over to any number of crystals. A real scanner, such as the GE Advance at the PET and Cyclotron Unit at Copenhagen university Hospital, has  $S = 672$  crystals.

The simplest PET-model assumes that a detected event occurred on the line that connects the center of two crystals. These lines are shown in fig. 3.8, and it should be clear that even a few crystals provides a large number of LORs.

**Figure 3.8:**  
Some lines connected by crystal pairs. Each line is a point in the sinogram, and the number of detected counts are considered to be the line-integral.



Using the convention in fig. 3.8, it is easy to determine the parameter of LOR in normal form. If an event is detected by the crystals  $s_1$  and  $s_2$ , and the scanner ring has unit radius, then we have:

$$(3.4) \quad \theta = \text{mod}(s_1 + s_2 + S/2, S)180^\circ/S$$

$$(3.5) \quad \rho = \begin{cases} -\cos(180^\circ(s_2 - s_1)/S) & \text{for } S/2 \leq s_1 + s_2 < S3/2, \\ +\cos(180^\circ(s_2 - s_1)/S) & \text{otherwise.} \end{cases}$$

Here  $\text{mod}(a, b)$  is the modulus function, i.e.  $0 \leq \text{mod}(a, b) = a + kb < |b|$  for some  $k \in \mathbb{Z}$ .

In practice, the field-of-view (FOV) limits  $\rho$  so only LORs near the center are considered. Converting to crystals using eq. (3.5), this limit may be specified by a constant,  $B$ , so we only consider LORs that satisfy:

$$(3.6) \quad S/2 - B \leq \text{mod}(s_2 - s_1, S) \leq S/2 + B$$

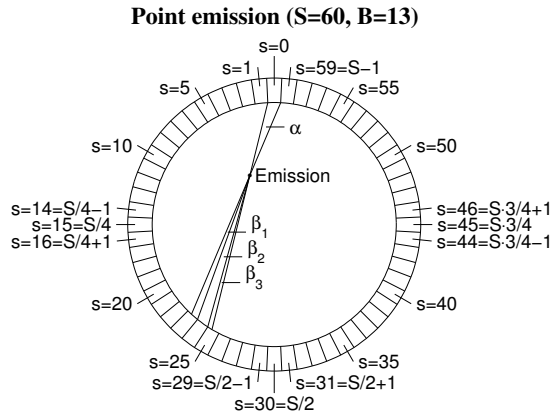
#### Sinogram bins

Just as for SPECT, the probability that an event is measured by a crystal pair is proportional to the solid angle subtended by the LORs that intersects the crystal pair and point of emission. Unlike SPECT, disregarding the 3D geometry is justified. If the scanner ring is thin, i.e. the distance from the crystals to the emission

point is large compared to the width of the ring, the angle in the  $z$ -axis is nearly constant.

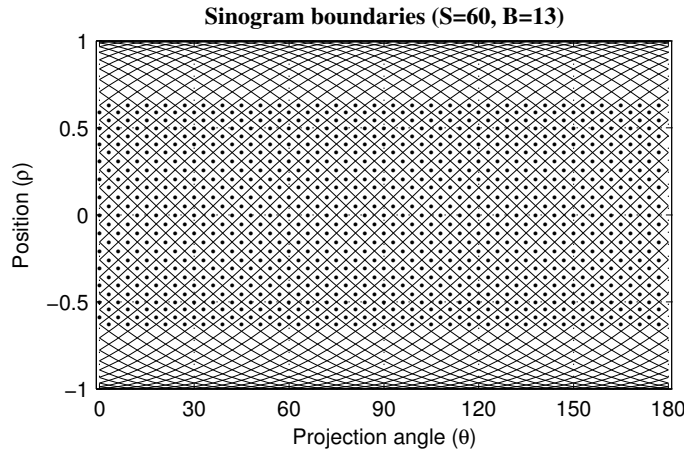
The geometry of a transaxial slice is illustrated in fig. 3.9. The probability that an emission intersects a given pair is proportional to the angle. In the current example, there are three pairs where the emission intersects crystal 0.

**Figure 3.9:**  
The probability of an ideal scanner detecting an annihilation is proportional to the angle. The probability that crystal 0 detects the event is proportional to  $\alpha$ , while the pair  $p(0, 23) \propto \beta_1$ .



In PET, the measured data is the number of counts detected by each pair of crystals (within the FOV). Thus, each event is put in a bin corresponding to an area of the sinogram. These bins are shown in fig. 3.10, together with the point corresponding to the LOR of the crystal centers.

**Figure 3.10:**  
Measured sinogram bins for an idealized PET-scanner, where all events are binned by the crystal pair intersected by the LOR. The dots represent the center lines used for a realistic FOV.



One thing evident in fig. 3.10 is that the bins do not correspond to equal areas. This is a consequence of the circular geometry, and can not be correctly classified as a flaw. Those who are mathematically inclined may view this as a consequence of eq. (3.5).

This fact is not too important when the FOV is restricted to small values of  $\rho$ . However, since most processing algorithms are designed for a uniformly sampled sinogram, it is customary to apply a *geometric correction* to the sinogram, which resamples the sinogram to take this into account.

**Model flaws**



A number of considerations have been ignored. As briefly mentioned in sec. 2.7, p. 18, the scintillation crystals in most PET-scanners are not single entities. Instead the ring consists of blocks with slit of different depths carved into it. The part of the block detecting an event is then inferred by interpolating the response of several photo-multiplier tubes on the back of each block.

This system is not perfect, and it is usually necessary to correct for the sensitivities of each block by calibration. Furthermore, many of the limitations mentioned in sec. 2.6.1 p. 17 are difficult to model accurately. Positron range can be taken into account, but dead-time can only be partially corrected.

### 3.6 Attenuation model

001010  
100001  
111100

As described in sec. 2.4 p. 12, some photons may be absorbed inside an object without reaching the scanner. The probability if this is an exponential function of the length traversed inside the object and the attenuation constant of the medium.

For brain scans, it is adequate to model a medium with a constant attenuation constant. Other scans that includes, say, airways and lungs, may not have this property. A further simplification in neurology is that transaxial cuts are generally convex (see e.g. p. 10). That is, if two points are inside the object, then any point on the connecting line-segment is also inside the object.

This implies that a line intersects the object boundary at exactly 0, 1, or 2 points. Now assume a LOR intersects the object as distances  $d_1 \leq d_2$  from the scanner, and the emission occurs at distance  $d_o$ . Then the length of the LOR that is inside the object is given by:

$$(3.7) \quad l = \begin{cases} 0 & \text{for } d_o \leq d_1, \\ d_o - d_1 & \text{for } d_1 < d_o < d_2, \\ d_2 - d_1 & \text{otherwise.} \end{cases}$$

The distances  $d_1$  and  $d_2$  may be precomputed at discrete points for each value of  $\theta$ . If the object is a polygon, these distances can be interpolated linearly from these samples. Once  $l$  is determined, the attenuation can be applied by applying eq. (2.1) p. 13.

In PET, the position of the object point is not needed. A point inside the object,  $d_1 \leq d_o \leq d_2$ , is detected when both photons reach the scanner which has the probability  $p = e^{-(d_o-d_1)\mu} e^{-(d_2-d_o)\mu} = e^{-(d_2-d_1)\mu}$ . Points outside the object also obey this law.

### 3.7 General linear models

∫∫∫

All the models above has one crucial property: the measured response is a linear function of the number of emissions at each point in the scanner. Each emission has a certain probability of being detected by a particular bin in the sinogram, but the number of detected events is linear in the number of emissions (ignoring non-linear properties such as dead-time).

Mathematically we may express this as follows:

$$(3.8) \quad \lambda \triangleq \mathcal{E}\{\mathbf{b}\} = \mathbf{A}\mathbf{x}$$

system matrix Here  $\mathcal{E}\{\cdot\}$  is the expectation operator, so  $\lambda$  is defined as a vector that contains the expected number of counts in each bin of the measured sinogram,  $\mathbf{b}$ . The number of events emitted from each point of the scanned image is represented by the vector  $\mathbf{x}$ , and they are related by the *system matrix*  $\mathbf{A}$ .

Generally, each element in  $\mathbf{x}$  represents a distributed area. In that case, each emission is assumed to occur from a point with a prescribed distribution. This distribution can also be modeled by  $\mathbf{A}$ .

All the discrete models in this chapter can be expressed as a system matrix. However, we stress that the vector  $\mathbf{x}$  represents an image in terms of a set of basis functions. This notion may be unfamiliar to programmers, who are accustomed to representing an image as a 2D-array (and a volume as a 3D array).

Thus, even if the continuous basis of the image is chosen to represent pixels (or voxels in 3D), it does not change the fact that  $\mathbf{x}$  is a vector. For simplicity, *all* vectors are column-vectors in this thesis. Thus, even if  $\mathbf{x}$  represents a rectangular image, it does not represent a matrix. Please refer to chapter 10 for a full discussion of this point.

hypercube To distinguish between vectors representing rectangular data and matrices, we shall use the term *hyper-cube* for a vector that represents an array. This has the advantage of being unambiguous (at least for the purposes of this thesis), and is not easily mistaken. It also conveys the important notion that the number of elements of the underlying data-array is irrelevant. In tomography, for instance, the FOV is typically defined by a circle or ellipse, and restricting an using a rectangular array to represent an image is inappropriate.

With this caveat out of the way, we shall nevertheless refer to elements of  $\mathbf{x}$  as pixels. While the underlying basis is mostly unimportant in this thesis, it remains helpful to use familiar terms. We refer to elements of  $\mathbf{b}$  and  $\lambda$  as bins for the same reason.

### 3.8 Noise model

∫∫∫

The linear model in eq. (3.8) only describes the average measurement  $\lambda$ . However, only a finite number of counts are measured, and this needs to be taken into account.

Generally, a very large number of radioactive nuclei are injected into the person being scanned. These are distributed spatially by a probability distribution; determining this distribution is the true objective of the scanning.

The probability that a nucleus emits a photon in a small volume is very small. Therefore, the number of photons emitted in such a volume is a stochastic variable,  $X$ , governed by the Poisson distribution discovered by Simeon Denis Poisson [60]:

$$P\{X = k\} = e^{-\lambda} \lambda^k / k!, \quad k \in \mathbb{N}$$

Here the parameter,  $\lambda$ , turns out to be both the mean and variance of  $X$ .

If each nucleus are considered to be distributed independently (i.e. a *tracer dose* of the substance is administered), then the number of photons emitted from the volumes are independent.

Each photon emitted from the volume has a fixed probability distribution of being counted by the bins of the sinogram. This distribution is exactly the models derived in this chapter. Thus, if we include an extra bin for undetected photons, the distribution of the bins is governed by a multinomial distribution.

It turns out that if the bins,  $b_1, b_2, \dots$ , are multinomial distributed with a parameter drawn from a Poisson distribution, then each of the bins are independent Poisson-distributed variables [45].

Thus, the number of counts detected by a bin,  $b_i$ , is a sum of independent Poisson-distributed variables. This, in turn, is merely another Poisson-distributed variable [60]:

$$(3.9) \quad P\{b_i = k\} = e^{-\lambda_i} \lambda_i^k / k!, \quad k \in \mathbb{N}$$

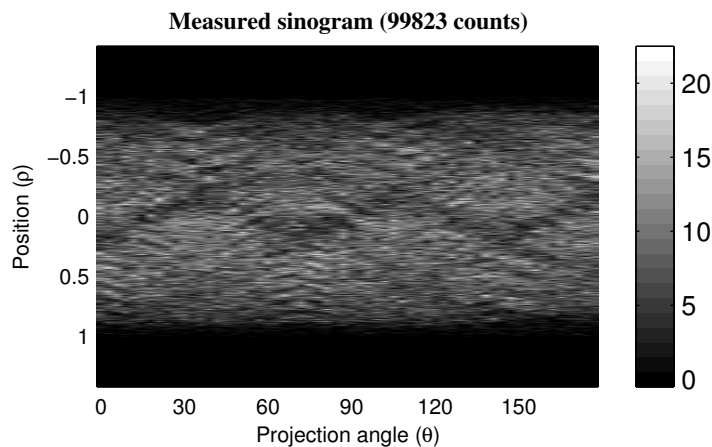
Note that the derivation showed that the measured counts are independent. This is a very powerful result, which will be used extensively.

In terms of linear models, we may express this as an error-term:

$$(3.10) \quad \mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{e} = \boldsymbol{\lambda} + \mathbf{e}$$

### Sinogram examples

We are now ready to show what the sinograms of the phantoms look like when noise is considered. For variety, the phantoms in fig. 3.4 and fig. 3.5 were scaled to incorporate about 100,000 (fig. 3.11) and 20,000 counts (fig. 3.12) respectively. Clearly, there is a substantial amount of information lost in the process.

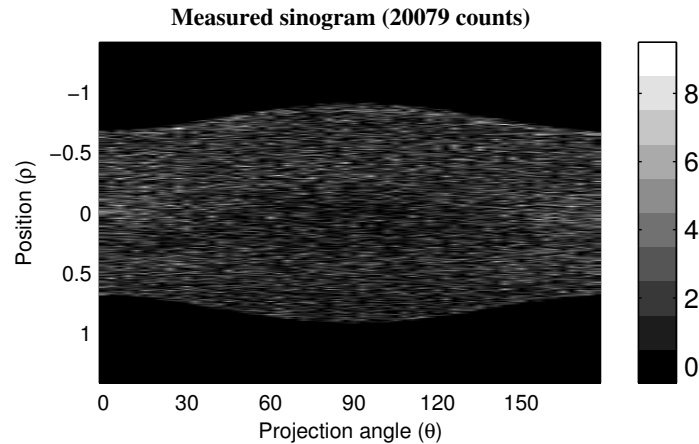


**Figure 3.11:**  
The sinogram in fig. 3.4 sampled from a Poisson distribution with approximately 100,000 counts.

### Model flaws

The Poisson-distribution is only valid when no amount of restoration has been performed on the sinogram. Thus, geometric correction, calibration of photomultipliers, linearization of dead-time, and other factors result in correlated measurements.

In principle, the Poisson-statistic could be maintained for linear operations by randomized sampling. However, the idea of applying even more noise to the measurement is not appealing.



**Figure 3.12:**  
The sinogram  
of the Shepp-  
Logan phantom  
in fig. 3.5 sampled  
with approximately  
20,000 counts.

At a more fundamental level, a PET-scan corrected for randoms using on-line subtraction can result in negative counts as described in sec. 2.6.1. This breaks fundamentally with the Poisson-distribution.

To model this more accurately, a shifted Poisson-distribution can be used [85]. This uses a Poisson-distribution of the combined variance, but shifted to match the average.

### 3.9 Software

```
001010
100001
111100
```

All phantoms in this thesis were created by `bbphantom` found in `BBTools`. The system-matrix corresponding to a discrete Radon-transform, using the algorithm in sec. 3.3.1, can be created as a black-box operator by `bbradon` (see chapter 10 for information about black-box operators).

A more elaborate model is available as `bbtomoemit`, which includes beam-divergence (sec. 3.4), and uniform attenuation of a convex body (sec. 3.6). Geometric correction is not currently implemented, but may be included by an interpolation.

The `MATLAB`<sup>®</sup> image processing toolbox from Mathworks includes `phantom`, which can create a Shepp-Logan phantom. Unlike `bbphantom`, however, the average value of the pixels depends on the image size. The function `radon` computes the Radon transform of an image, but the corresponding back-projection is not included.

More elaborate system-matrices can be generated by the package `ASPIRE` from Fessler [23]. He also distributes an image reconstruction toolbox for `MATLAB`<sup>®</sup> that contains simple implementations of the Radon transform.

Toft showed in [72] how to implement a high-speed Radon-transform using slant-stacking. He also distributes implementations of the algorithms along with his thesis.

A different approach is to use Monte-Carlo simulations, although these are mostly useful for validation purposes. One of the most popular packages is `SimSET` (Simulation System for Emission Tomography) [75], but a number of packages are freely available.

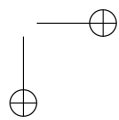
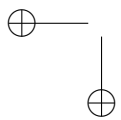
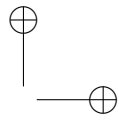
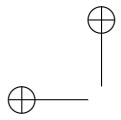
### 3.10 References

The Radon transform is treated analytically in a tomographic setting by a number of authors. One of the most complete is the thesis by Peter Aundal Toft [72], which derives the Radon transform of a number of analytical functions.

Both the SPECT- and PET-models were derived from first principles, and more elaborate models is tantamount to instrumentation models. We refer to sec. 2.9 p. 19 for references.

Linear models and the Poisson statistic are used throughout the reconstruction literature. In particular we highlight [45], as it is both clear and includes a list of properties of the Poisson distribution.

Fessler argued that samples of a Gaussian distribution could be a better noise-model than the Poisson-distribution on PET-systems with on-line subtraction [20]. A more comprehensive analysis of non-Poisson distributed counts and corresponding likelihood-functions is the thesis of Yavuz [84], which advocates the use of the shifted Poisson-distribution.



# Classical reconstruction

*This chapter goes through the classical reconstruction algorithms used in emission tomography. We start with a description of back-projection, which is a basic building-block also used in other part of the thesis.*

*This is followed by a detailed description of the two methods implemented on most scanners: FBP (Filtered Back-Projection) and EM (Expectation Maximization). We give examples of images reconstructed with each of these methods.*

*For reference, we also mention a few other methods found in the litterature.*

## 4.1 Methods and abbreviations

As usual, there are a plethora of abbreviations found in the litterature. The abbreviations most used in the literature are shown in table 4.1. The first part are common, whereas the lower part as encountered more rarely.

Abbreviation	Meaning
FBP	Filtered Back Projection
ML	Maximum Likelihood
EM	Expectation Maximization
OSEM	Ordered Subset EM
ART	Algebraic Reconstruction Technique
WLS	Weighted least-squares
PWLS	Penalized weighted least-squares

**Table 4.1:**  
*Terms used in classical reconstruction methods.*

## 4.2 Back-projection

The back-projection is used in some form in most reconstruction algorithms. Although it is rarely used by itself, we shall study it in some detail before proceeding to more sophisticated algorithms.

Back-projection is a one-liner: the counts detected in each bin are distributed evenly on the corresponding LOR. If we consider a point-source, all LORs intersect the point and the total length of LORs in a small area decreases with the distance from the point.

∫∫∫

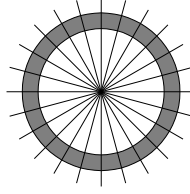
### Intensity function

Consider an annulus around the emission-point as shown in fig. 4.1. If the radii of the annulus are given by  $r_1 < r_2$ , then the area of the annulus is  $A = \pi(r_2^2 - r_1^2)$ . If the sinogram contains  $P$  samples of  $\theta$  (i.e.  $P$  LORs intersects the point of emission), then the total length of LORs in the annulus is  $2P(r_2 - r_1)$ .

This means that the intensity of the annulus is given by:

$$(4.1) \quad \frac{2P(r_2 - r_1)c}{\pi(r_2^2 - r_1^2)} = \frac{2Pc}{\pi(r_1 + r_2)} \rightarrow \frac{Pc}{\pi r_1} \text{ for } r_2 \rightarrow r_1.$$

### Back-projection



**Figure 4.1:**  
An annulus around  
a point-source and  
the intersecting  
LORs.

Here  $c$  is a scale-factor that determines the number of counts deposited on a LOR of unit length.

For completeness, the analogously result for a spherical shell for a 3D back-projection is:

$$(4.2) \quad \frac{3Pc}{2\pi(r_1^2 - r_1r_2 + r_2^2)} \rightarrow \frac{3Pc}{2\pi r_1^2} \text{ for } r_2 \rightarrow r_1.$$

If the LORs are distributed evenly (i.e.  $\theta$  is sampled equidistantly), then any area (or volume) inside the annulus (or shell) will have approximately the same intensity. Therefore, the reconstructed point will result in a symmetric point-spread function in this case.

### Discrete back-projection

For a discrete system, the forward projection of bin  $i$  is given by the linear equation  $\lambda = \mathbf{A}\mathbf{x}$ :

$$(4.3) \quad \lambda_i = \sum_j \mathbf{A}_{(i,j)} \mathbf{x}_j$$

To back-project  $\lambda_i$  or  $\mathbf{b}_i$ , we must distribute the counts to  $\mathbf{x}$  in proportion to the amount (“LOR-length” for a Radon transform) each element contributed to the measurement. Specifically, the back-projection is:

$$(4.4) \quad \hat{\mathbf{x}}_j = \sum_i \mathbf{A}_{(i,j)} \mathbf{b}_i$$

More simply, this can be written as  $\hat{\mathbf{x}} = \mathbf{A}^T \mathbf{b}$ .

001010  
100001  
111100

### Implementation

Scaling can not be ignored in a real implementation. The simplest way to deal with scaling is to distribute each bin evenly along the LOR which is inside the field of view. Note, however, that a point-source will no longer be reconstructed as a symmetric function.

To this end, consider eq. (4.4). If we distribute the counts in  $\mathbf{b}_i$  according to the weights given by  $\mathbf{A}_{(i,:)}$ , then we need to scale by the sum of the weights:

$$(4.5) \quad \hat{\mathbf{x}}_j = \sum_i \mathbf{b}_i \mathbf{A}_{(i,j)} / \sum_k \mathbf{A}_{(i,k)}$$

This assumes that  $\mathbf{A}$  is non-negative, i.e. there are no negative elements of  $\mathbf{A}$ . All models described in chapter 3 satisfy this requirement.



With this scaling, the number of counts in  $\hat{\mathbf{x}}$  matches the number of counts in  $\mathbf{b}$ . However, we prefer images which corresponds to image originally put in the scanner. The Radon-transform and, by extension, all other models in this thesis, project a point of unit intensity to one count for each projection. Thus, if we sample  $\theta$  at  $P$  discrete positions, the reconstructed point will contain  $P$  counts. Thus, if we divide by  $P$ , we get a scale corresponding to the original image.

This leads directly to alg. 4.1. Since this is the first algorithm in this thesis, a few comments are in place:

- The result is an operator which performs the back-projection. To back-project a specific sinogram,  $\mathbf{b}$ , the matrix-vector product  $\hat{\mathbf{x}} = \mathbf{B}\mathbf{b}$  must be computed, i.e.  $\mathbf{x}_h = \mathbf{B} * \mathbf{b}$ .
- The algorithm use BBTools described in chapter 10. The function `bbdiag` is a black-box implementation of the MATLAB<sup>®</sup>-function `diag` and functions in much the same way. We will assume that functions in BBTools similar to MATLAB<sup>®</sup> can be used without explanation.
- Generally, the algorithms in the thesis are meant to be illustrative. However, since the system-matrix  $\mathbf{A}$  can be a general black-box implemented as compiled mex-files, the operator produced by alg. 4.1 is competitive with any other algorithm based on compiled code.

Those who are theoretically inclined may find that the syntax  $\mathbf{x}_h = \mathbf{B} * \mathbf{b}$  is merely cosmetic.

### 4.3 Examples of back-projection

With this implementation in mind, we may show the result of the back-projection algorithm. For convenience, the original phantom, fig. 3.1, is reproduced in fig. 4.2.

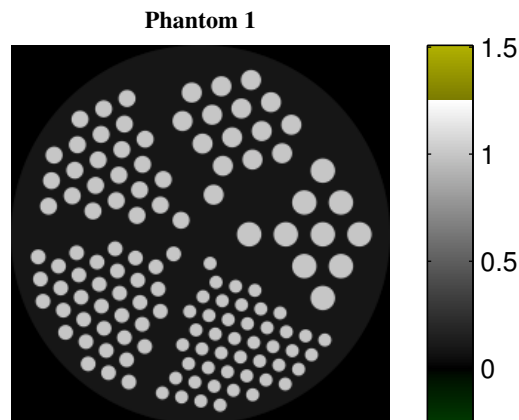
It should be evident that pure back-projection is inadequate as a reconstruction algorithm. It is difficult to discern even the largest hot-spots in the phantom and edges are not preserved well.

**Algorithm 4.1: Back-projection**

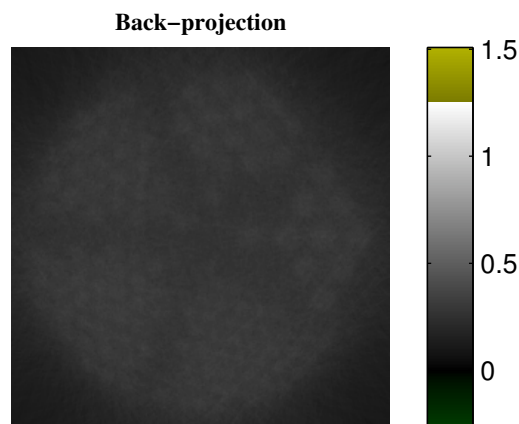
*Create a back-projection operator given a non-negative system-matrix,  $\mathbf{A}$ , and the number of projections.*

```
function B = backproject(A, P)
    p = A * ones(size(A, 2), 1);           Project to determine weights
    c = double(p > 0);                    Zero out null entries
    p(p == 0) = 1;                         Avoid division-by-zero
    B = AH * bbdiag(c ./ (P * p));       Construct operator
```

**Figure 4.2:**  
A reproduction  
of the resolution-  
phantom in fig. 3.1.



**Figure 4.3:**  
The resolution-  
phantom in fig. 4.2,  
reconstructed from  
the noisy sinogram  
in fig. 3.11 using  
back-projection.



## 4.4 Filtered back-projection

The Filtered Back Projection algorithm (FBP) the dominant algorithm used in medical reconstruction. The images produces are linear in the inputs, and the algorithm is sometimes known simply as “linear reconstruction”. As the name suggest, it is based on back-projection, but use a filter to compensate for the weakness of pure back-projection.

Again, the underlying idea is simple: solve the linear system  $\mathbf{Ax} = \boldsymbol{\lambda}$  for  $\mathbf{x}$ , where  $\mathbf{A}$  is the discrete Radon-transform. Unlike back-projection, however, finding an efficient way to implement the scheme is not intuitive.

∫∫∫

### Derivation of FBP

Filtered back-projection is usually derived in the continuous domain using the *Fourier Slice Theorem* (also known as the *Central Slice Theorem*). The details are messy, and require careful attention at singularities. Since most of this thesis is based on linear algebra, we shall give an informal derivation that lends itself to an implementation in this setting.

The starting-point for FBP is a direct solution of the linear system  $\boldsymbol{\lambda} = \mathbf{Ax}$  which, in the presence of noise, becomes  $\mathbf{Ax} \approx \mathbf{b}$ . As mentioned previously, the solution

is given by a filter followed by a back-projection:

$$(4.6) \quad \hat{\mathbf{x}} = \mathbf{A}^T \mathbf{F} \mathbf{b}$$

We may now plug  $\hat{\mathbf{x}}$  into the linear equation:

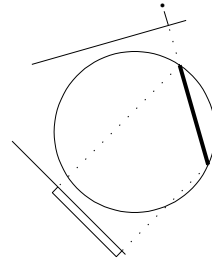
$$(4.7) \quad \mathbf{b} \approx \mathbf{A} \hat{\mathbf{x}} = \mathbf{A} \mathbf{A}^T \mathbf{F} \mathbf{b}$$

Thus, a good filter approximates the inverse (or pseudo-inverse in the general case) of the operator  $\mathbf{A} \mathbf{A}^T$ . In other words, if a sinogram with a single non-zero bin is filtered and  $\mathbf{A} \mathbf{A}^T$  is applied, then the new sinogram should be approximately the same as the starting point.

Consider  $\mathbf{A} \mathbf{A}^T$ , which represents a back-projection followed by a forward projection. Assume the system-matrix  $\mathbf{A}$  is the Radon-transform, and let us momentarily assume that only  $\theta$  is discrete.

The situation for a circular FOV is sketched in fig. 4.4. A bin is first distributed on the LOR, which is in turn forward projected to the same point. On the other hand, when the LOR is projected on a plane with a different angle, the result is a rectangular function.

Illustration of  $\mathbf{A} \mathbf{A}^T$



**Figure 4.4:**  
Back-projection  
followed by a forward  
projection,  
representing the  
operator  $\mathbf{A} \mathbf{A}^T$ .

Since the projection may be placed anywhere, we need a position-invariant filter that blocks rectangular functions. Such a filter is most easily designed in the frequency domain. If  $r(x)$  is a rectangular function of width  $2a$ , symmetric around  $x = 0$ , then the Fourier-transform,  $R(\omega) = \mathcal{F}\{r(x)\} = \int_{-\infty}^{\infty} r(x) e^{-j\omega x} dx$ , is given by [86]:

$$(4.8) \quad R(\omega) = \begin{cases} 2a & \text{for } \omega = 0, \\ 2 \sin(a\omega)/\omega & \text{otherwise.} \end{cases}$$

This can not be completely suppressed, but it can be rendered mostly harmless by damping low frequencies. In fact it turns out that it suffice to neutralize the denominator  $\omega$  by applying a ramp-filter:

$$(4.9) \quad F(\omega) = \begin{cases} |\omega| & \text{for } |\omega| < \omega_c, \\ |\omega|/2 & \text{for } |\omega| = \omega_c, \\ 0 & \text{otherwise.} \end{cases}$$

The cut-off frequency,  $\omega_c$ , is a necessary for the inverse Fourier-transform of  $F(\omega)$  to exist [44].

The inverse Fourier-transform of  $R(\omega)F(\omega)$  has sharp peaks near  $x = \pm a$ , but the integral of an area around a peak is small and diminishes as  $\omega_c$  becomes large.



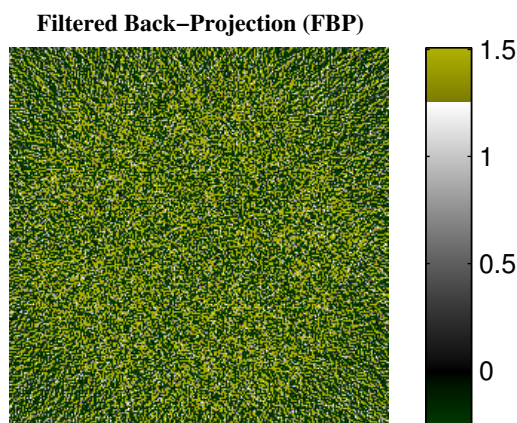
- The operator produced by alg. 4.2 is an ideal *pre-conditioner* for an iterative algorithm that solves a system using a more complex model than the Radon transform.

For example, assume that  $A$  is a model that includes beam-divergence and attenuation,  $B$  is the FBP-operator constructed from the Radon transform, and  $b$  is the measured sinogram. Then a standard iterative solver, e.g.  $xh = \text{lsqr}(B^*A, B^*b)$ , converges in a few steps.

## 4.5 Examples of FBP

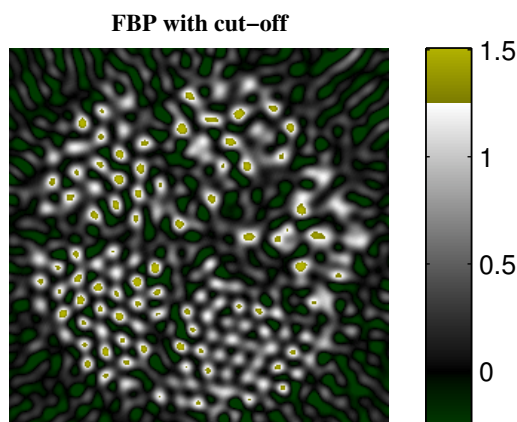
The basic operator created by alg. 4.2 was used to reconstruct the resolution-phantom using the noisy sinogram with 20,000 counts. The result is shown in fig. 4.5, which is clearly a useless approximation of the original phantom in fig. 4.2.

**Figure 4.5:**  
The resolution-phantom reconstructed using FBP with cut-off  $\omega_c = 1$ .



To get a more useful image, we need to filter out the noise by lowering the cut-off frequency. Exactly what value to use depends on the circumstances and intended use of the image, but as we only need to illustrate the technique, we will use the arbitrary value of  $\omega_c = 0.175$ . The result is shown in fig. 4.6.

**Figure 4.6:**  
A new reconstruction, but low-pass filtered with a cut-off  $\omega_c = 0.175$ .



Note that smoothing comes with a loss of resolution. The ultimately smooth image (cut-off  $\omega_c = 0$ ) is blank and contains no information.

An artifact often seen in reconstructed images is “ringing”. In this context it usually manifests itself as streaks radiating from the center and outwards. This is sometimes called “star artifacts” [19].

The term “ringing” comes from electronics and is meant in the sense of a bell-ringing. Here, it comes from the abrupt change at  $\omega_c$  in eq. (4.9), which is converted to a highly oscillating function. As touched briefly in sec. 4.4.1, the average is negligible, but the peaks will always be high. This is a well-known result known as Gibbs phenomenon [54].

To reduce this kind of artifacts, it is common to modify the filter by multiplying by a window-function. It may be confusing, because there are plenty to choose from in the literature. In practical terms this is not too important, and the author<sup>1</sup> is inclined to agree with the following quote [58, p.534]:

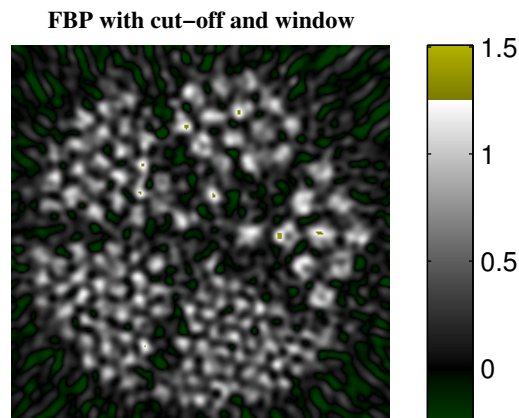
There is a lot of unnecessary lore about choice of a window function, and practically every function that rises from zero to a peak and then falls again has been named after someone.

...

However, at the level of this book, there is effectively *no difference* between any of these (or similar) window functions.

We shall stick to the so-called Hamming window below. One thing to notice about windows is that it is often possible to increase the cut-off frequency. Fig. 4.7 below was reconstructed with  $\omega_c = 0.3$  to yield approximately the same resolution as fig. 4.6.

**Figure 4.7:**  
Another reconstruction, low-pass filtered with cut-off  $\omega_c = 0.3$  but including a Hamming window to reduce star-artifacts.



## 4.6 EM reconstruction

After back-projection and, possibly, attempts at direct inversion of the system-operator, the EM-algorithm is likely to be one of the first algorithms a novice would think of. To back up this claim, the author knows of 4 persons who came up with the idea independently<sup>2</sup>.

<sup>1</sup>The author is an engineer formally trained in the “lore”.

<sup>2</sup>This count includes the author.

Let us rehash the idea of back-projection: the counts measured in a bin is distributed evenly on the LOR from which they could have come. Now suppose we already have a rough idea about what the image looks like. Instead of distributing the counts evenly, we may distribute them on the LOR in proportion to the concentrations in the approximate image.

This is the basic step of the EM-algorithm, and it produces an estimate which should be better than the image we started with. Thus, we may improve the new image by applying a new step. Thus we have an iterative procedure that, hopefully, converges to a fix-point.

The EM-algorithm is sometimes known as “iterative reconstruction” (as opposed to “linear reconstruction”), because it is the second major algorithm implemented on many commercial scanners.

001010  
100001  
111100

### Implementation

As explained, the counts in bin  $b_i$  should be distributed in the LOR in the proportion given by the pixels in the previous estimate,  $\hat{x}_j$ , in the LOR. Since the length of the LORs are represented by a system-matrix, the counts must also be weighted with the elements of  $\mathbf{A}$ . We may therefore express the EM-step by an equation similar to eq. (4.5):

$$(4.10) \quad \hat{x}'_j = \sum_i b_i \mathbf{A}_{(i,j)} \hat{x}_j / \sum_k \mathbf{A}_{(i,k)} \hat{x}_k$$

Like the back-projection algorithm, this produces an image with a total number of counts that matches the number of acquired counts, and we must divide by the number of projections to get an image comparable with the original.

There is, however, a more general way to handle this. Instead of dividing by the number of projections, we can divide each pixel by the amplification factor:

$$(4.11) \quad \begin{aligned} \hat{x}'_j &= \frac{1}{\sum_i \mathbf{A}_{(i,j)}} \sum_i b_i \mathbf{A}_{(i,j)} \hat{x}_j / \sum_i \mathbf{A}_{(i,j)} \hat{x}_j \\ &= \left( \frac{\hat{x}_j}{\sum_i \mathbf{A}_{(i,j)}} \right) \sum_i \mathbf{A}_{(i,j)} \left( \frac{b_i}{\sum_k \mathbf{A}_{(i,k)} \hat{x}_k} \right) \end{aligned}$$

This is the final iteration of the EM-algorithm, and the bulk of the work can be performed efficiently as matrix-vector products as shown in alg. 4.3.

### Ordered subset

The EM-algorithm converges rather slowly, and since it is used in practical work it is desirable to speed it up. The *ordered subset* algorithm, invented by Harold Malcolm Hudson and Richard S. Larkin [37], is one way implemented on many scanners, often under the name OSEM.

The idea is to update only a subset of the projections at each iteration. We shall not discuss this algorithm further, as it is merely intended to be a faster version of the EM-algorithm.

∫∫∫

### Justification of EM

The idea described above is simple to understand, but there was little to motivate it and even a cursory analysis reveals problems. Suppose we draw the LORs corresponding to all bins with non-zero counts. If we create an iterate by placing a single point on each LOR at random, except where it intersects another LOR, then it converges to a fix-point in one step.

While this result appears alarming, we note that it may be impossible to construct such a solution in a discretized image. It does, however, suggest that algorithms that are not backed by solid theory should be regarded with suspicion.

Happily, the simple iteration described above happens to coincide with an *Expectation Maximization* (EM) algorithm, which computes a maximum-likelihood solution for a likelihood defined by the Poisson-distribution.

It is beyond the scope of this text to delve in the theoretical foundation of this algorithm and refer to the references in sec. 4.9 for details. We will, however, state the basic result about convergence.

**Theorem 4.1: (EM-Convergence)** *Let  $\mathbf{b}$  be a sample of random vectors with independent Poisson-distributed elements with parameters  $\boldsymbol{\lambda} = \mathbf{A}\mathbf{x}$ . Furthermore, let  $\hat{\mathbf{x}}_0$  be an image with entirely positive pixels.*

*Assume that  $\mathbf{A}$  is non-negative, and that no element in  $\mathbf{A}^T\mathbf{b}$  is zero, i.e. all pixels in the image are intersected by a LOR corresponding to a bin in  $\mathbf{b}$  with non-zero counts. Finally, assume that  $\text{null}\{\mathbf{A}\} = \{\mathbf{0}\}$ .*

*Then the iterates of the EM-algorithm,  $\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_1, \dots$ , converges to the maximum-likelihood solution:*

$$(4.12) \quad \lim_{i \rightarrow \infty} \hat{\mathbf{x}}_i = \underset{\mathbf{x}}{\text{argmax}} \text{Prob}(\mathbf{b} | \boldsymbol{\lambda} = \mathbf{A}\mathbf{x})$$

PROOF

See [45]. ■

The most revealing about this theorem is the conditions required to make it true. First, the number of pixels in the image must necessarily be smaller than the number of sinogram bins. Secondly, sampling the sinogram denser to accommodate

**Algorithm 4.3: EM-Reconstruction**

*Reconstruct an image using the EM-algorithm, given a non-negative system-matrix,  $\mathbf{A}$ , a measured sinogram of independent Poisson-distributed counts,  $\mathbf{b}$ , and a limit on the number of iterations.*

---

```

function  $\hat{\mathbf{x}} = em(\mathbf{A}, \mathbf{b}, i_{max})$ 
   $\mathbf{w} = \mathbf{A}^T * ones(size(\mathbf{A}, 1), 1);$            Back-project to determine weights
   $\mathbf{w}(\mathbf{w} \equiv 0) = 1;$                        Avoid division-by-zero
   $\hat{\mathbf{x}} = ones(size(\mathbf{A}, 2), 1);$              Initialize iterate
  for  $i = 1 : i_{max}$ 
     $\hat{\mathbf{x}}_p = \hat{\mathbf{x}};$                        Save iterate for convergence test
     $\mathbf{p} = \mathbf{A} * \hat{\mathbf{x}};$                    Forward-project current estimate
     $\mathbf{p}(\mathbf{p} < \epsilon_M) = \epsilon_M;$          Avoid division by zero
     $\hat{\mathbf{x}} = (\hat{\mathbf{x}} ./ \mathbf{w}) .* (\mathbf{A}^T * (\mathbf{b} ./ \mathbf{p}));$  Evaluate eq. (4.11)
    if  $\|\hat{\mathbf{x}}_p - \hat{\mathbf{x}}\|_2 < \|\hat{\mathbf{x}}\|_2 * 1e - 3,$  break Stop if we reached a fix-point
  end for

```

---



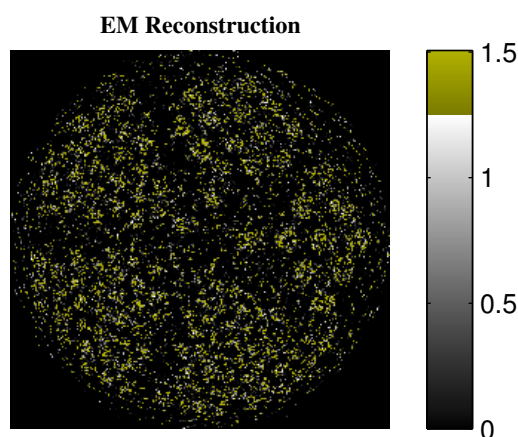
more pixels in the image is only helpful if the number of acquired counts is large enough to make most of them non-zero.

In other words, the EM-algorithm relies on *pixelation* of the image to function. The EM-algorithm must be able to “see” the pixels, i.e. the images produced by the method are not an approximation of an underlying continuous function. This is a distinct disadvantage.

## 4.7 Examples of EM

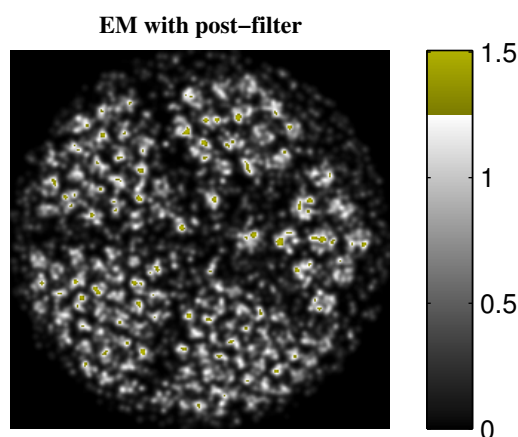
If the EM-algorithm is used naively it often produces useless results. Fig. 4.8 shows the resolution-phantom reconstructed using the EM-algorithm. Although better than an unfiltered version of a FBP-reconstruction, it is not useful when used in this form.

**Figure 4.8:**  
The resolution-phantom reconstructed using the EM-algorithm.



One way to produce a useful image, is simply to apply a post-filter on the reconstructed image. The image in fig. 4.9 is the same as fig. 4.8 after a Gaussian filter with a FWHM of about 3.52 was applied. It is clear that such a filter is necessary.

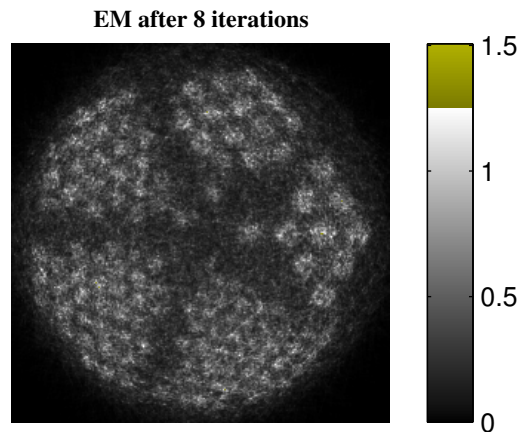
**Figure 4.9:**  
Fig. 4.8 post-filtered with a Gaussian kernel with FWHM  $\approx$  3.52 pixels.



Another popular way to avoid excessive noise, is early stopping. This simply put a limit ( $i_{max}$  in alg. 4.3) on the number of iterations taken and prevents the EM-algorithm from converging to the maximum-likelihood solution. The result after 8

iterations is shown in fig. 4.10. Early stopping is another way to speed up the EM-

**Figure 4.10:**  
EM-reconstruction  
using early stop-  
ping.

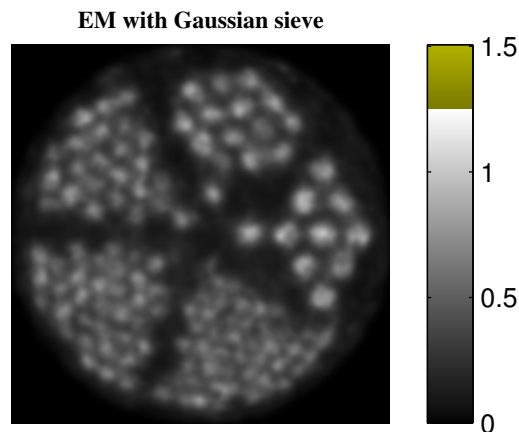


algorithm, since the number of iterations are limited to a fixed number. However, each iteration still require approximately twice the work of the FBP-algorithm, and can not be recommended in general.

A better way to incorporate a filter is to use a *sieve* [65]. In the elementary form, each iteration in the EM-algorithm is followed by a Gaussian filter. This filter should have a substantially smaller kernel than a post-filter, and significantly reduces the number of iterations required for convergence.

The resolution-phantom reconstructed using a sieve is shown in fig. 4.11. Given the speed of this algorithm, together with the solid underlying theory, this should method can generally be recommended.

**Figure 4.11:**  
EM-reconstruction  
using a Gaussian  
convolutive sieve  
with FWHM $\approx$  1.2  
pixels.



## 4.8 Other algorithms

Historically, the first algorithm was Algebraic Reconstruction Technique (ART), which is a direct solution of the linear system using a Gauss-Seidel iteration. There are several similar algorithms, e.g. SIRT (simultaneous iterative reconstruction technique), SART (simultaneous algebraic reconstruction technique), and Kaczmarz method. All of these are obsolete, and should not be used any more.

An alternative to filtered back-projection is “filtering after back-projection”. The method applies a filter after back-projection to compensate for the spread given by eq. 4.1. Unlike FBP, the problem with singularities is easily circumvented in the discrete case by integrating over a pixel or putting the singularity between pixel-centers.

## 4.9 References

The original paper by Radon [59] is in the domain of people who cope with Stieltjes-, Cauchy-, and ordinary Riemann-integrals. A fair knowledge of German is recommended, but it is a worthy read for anyone demanding mathematical rigor. Many equations found in the literature can be found here together with the assumptions necessary for them to be valid; a point often glossed over elsewhere.

Traditional derivations of filtered back-projection, based on the Fourier slice theorem, can be found in numerous places. A terse derivation is given in [5], while a more verbose version can be found in [72]. While such derivations may appear to motivate the ramp-filter, they do, in the opinion of this author, fall short in rigour. If rigor is desirable, the paper by Radon should be consulted.

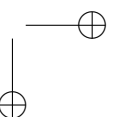
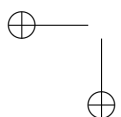
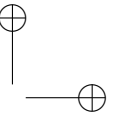
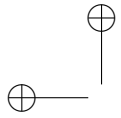
The treatment of FBP given here is new, but the interpretation as an inverse of the operator  $\mathbf{A}\mathbf{A}^T$  was noted by Kawata and Nalcioqli [40]. However, they made no attempt at analyzing the operator other than stating that the inverse “corresponds to the two-dimensional rho-filter”.

EM-algorithms are effective at finding maximum-likelihood solutions for a broad class of problems, and several excellent texts, such as [8], are available. The original papers in the context of reconstruction is by Shepp and Vardi [64], and Lange and Carson [45]. Of these, the latter should be preferred as it includes scaling and the convergence-proof in theorem. 4.1.

The method of sieves was first used for tomographic reconstruction by Snyder and Miller [65]. The paper includes a clear explanation of the problem with the basic EM-algorithm and can be recommended for this reason alone. It also argues that pixelation, in isolation, does not help the EM and establish that Gaussian sieves are backed by solid theory.

Eugene Veklerov and Jorge Llacer took the sieve method a step further in [76], where they gave a rigorous method for choosing the sieve in the EM-algorithm.

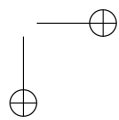
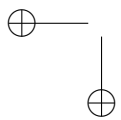
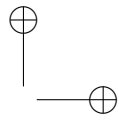
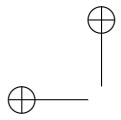
A classical reference for the early methods described in sec. 4.8 is [39], while “filtering after back-projection” is described in a continuous setting in [72].





## Part II

# Improved Reconstruction



# Reconstruction goals and strategies

*This chapter contains a discussion about the practical requirements imposed on a reconstruction algorithm. The majority of this chapter is collected from personal experience, and is therefore less rigorous than other parts of the thesis. Many of the opinions expressed in this chapter are personal opinions of the author, and should not be construed as being universal.*

*When other requirements than accuracy are considered, linear methods have many desirable properties that are important from a user-point of view. We discuss previous research in this context, and go forward with a description of regularized reconstruction.*

## 5.1 User requirements

One truth holds in engineering as well as other aspects of life: if you can not formulate what you want, then the chance of getting it is remote. The usually way for engineers to express “what they want” is through a cost-function. When a cost-function has been decided upon, a solution can be quantitatively evaluated and therefore optimized.

It follows that it is important to consider the properties of the cost-function. Usually it is necessary to balance several conflicting goals, and this is especially true when it comes to reconstruction. Since the work in this thesis aims to provide useful solutions, the demands of the users should be considered.

Many issues are important from a clinical point of view. Although some requirements may be relaxed in research studies, it still remains that methods with little clinical prospect are unlikely to be used much. The following list is a compilation of issues drawn from the author’s experience from working with potential users.

1. **Low variability:** If a person is scanned twice the results should be alike. Scans between persons must also look the same, as this is the basis that clinicians use for diagnosing illness by inspecting the images.
2. **Minimal clinical work:** The system must be useful without demanding much extra clinical work. For instance it is not reasonable to assume generally that several MR-images are available.
3. **Minimal manual processing:** The level of required (i.e. mandatory) manual work on the collected data must be low.
4. **Consistent with old methods:** If at all possible, the reconstruction should allow a gradual introduction of new features. This allows clinicians to start with something familiar and introduce more advanced methods as they expand their understanding of the system.

5. **Homogeneous:** It greatly facilitates interpretation of the images if they are homogeneous, i.e. the point spread function (PSF) should be the same over all parts of the image. For specificity, we shall use the term “homogeneous” if the PSFs are spatial invariant and isotropic, i.e. independent of position and direction.
6. **Intuitive:** The reconstructed image and the true concentration should be related in an “obvious” way. If the reconstruction is inhomogeneous, it should at least be possible to characterize it accurately in a form which can be conveyed to the user in a comprehensible form.
7. **Free of artifacts:** While noise is unavoidable and images will never be perfect, a good reconstruction algorithm should suppress artificial artifacts. In other words, noise must not fool a reconstruction algorithm to produce an image with brain-like features.
8. **Few assumptions:** Clinicians who uses the images for diagnosing patients can be uneasy with methods based on elaborate models of the brain. Their interests are mainly parts of the brain that are abnormal or lesioned, and it is hard to build a prior that is acceptable to them.
9. **Coverage:** It should be possible to use the method for most scannings seen by an individual doctor. As an example, an algorithm should not build on assumptions proven only for a single tracer. The training necessary for interpreting images reconstructed with a new method is not justified unless it can cover a reasonable amount of the total number of scannings.
10. **Comparable:** In dynamic studies the different time frames must be reconstructed similarly, to make it possible to interpret the sequence. For this reason the FBP-algorithm is currently preferred to EM in such studies.
11. **Numerically robust:** The method should always work. The users are trained to understand human physiology, not numerical algorithms. The user will generally be unable to intervene if the reconstruction misbehaves.
12. **Rigorous and simple:** Engineers, physicists, and mathematicians are generally easier to convince than clinicians when it comes to declaring one method better than another. Nevertheless it remains helpful to use well known mathematical theories. It is important that this group “likes the method”, since the clinicians are unlikely to even consider a method unless the other experts agree.
13. **Quantitative and accurate:** Reconstructed images should reflect the concentration distribution as closely as possible. The ability to compute ratios between concentrations in various regions of the brain has a big impact on both diagnostics and research. In particular the ability to model tracer-kinetics depends on this.

## Discussion

A project that focuses on accuracy alone is not likely to have much success; it was no mistake that this item is last on the list. Even though accuracy is perhaps the easiest requirement to put into a cost-function, we should not forget that the



people who will use the images are trained at interpreting them. This means that consistency between images are usually a more important parameter.

Looking at the list as a whole, one may get the impression that clinical users are overly conservative. Generally, new methods are introduced only if the diagnostic power is demonstrably improved. To illustrate this point, we mention that it is still subject to debate whether attenuation-correction should be used in clinical SPECT. If users could be convinced by, say, a root-mean-square metric, this question would have been settled a decade ago. In 2002, only 9 of 19 Danish SPECT-centers used it routinely<sup>1</sup>.

The ability to produce results similar to existing methods can not be underestimated. In engineering, this goes under the heading “backwards-compatibility” and is by no means limited to reconstruction algorithms. Consider the likely success of the following products, regardless of technical merits:

- A text-processor that can not read any existing document.
- A CPU unable to run any previously purchased software.
- A car running on an exotic fuel, but neither gasoline nor diesel fuel.

It should be noted that lives are potentially at stake. A method should reflect this in the sense that *if* it delivers an answer it better be a good one.

A method should be viewed in light of the discussion above, although a research project can choose to ignore some aspects in order to set the stage for progress. However, principles that inherently fails to balance the issues are unlikely to be adopted for medical use.

## 5.2 Technical requirements

There are also a number of technicalities that should be considered. Unlike the user requirements, these are relatively easy to address.

### Computational power

Computational power represents a one-time (possibly large) investment but not clinical work. It is reasonable to expect that such investments will be made if they offer significant advantages over old methods.

Another factor is the continued increase in computational power. Any method that runs on existing hardware is likely to be useful for routine use on computers developed in 5 years. According to the industry’s own predictions [63], based on factories under construction and laboratory technologies, the current trend is expected to continue to at least 2016.

<sup>1</sup>From the seminar “New challenges in SPECT”, held by The PET and Cyclotron Unit, Copenhagen University Hospital, 19–20 Sep. 2002.

## Model mismatch

It should be evident from chapter 3 that scanning models used in reconstruction are somewhat crude approximations. This is a result of the complicated interactions in the scanning process, uncertainties in calibrations, and approximations necessary to make the reconstructions practically feasible.

However, there is another source of model mismatch. The user may deliberately turn off some levels of realism for a number of reasons. For example, a model of scatter or attenuation could be turned off in order to approximate the usual results, to increase the reconstruction speed, or simply distrust.

There are situations where each of these reasons are perfectly valid. It is important to use methods that do not assume that the models are the actual truth, and robustness with respect to the model is very desirable.

## Non-negativity

Since concentrations are non-negative it is a question whether the reconstructed image should be allowed to contain negative values. This may also be posed as a question about whether it is acceptable to introduce bias to reduce variance. No reasonable<sup>2</sup> method can guarantee that a region, which is truly zero, will be reconstructed as zero. Thus, a non-negativity constraint implies that such regions will be biased.

From a technological point of view one may ask whether it is reasonable to expect non-negative concentrations when corrections result in negative counts, as discussed in sec. 2.6.1 p. 18.

Neither of these arguments are likely to convince a practitioner of the plausibility of negative concentrations. However, negative values occur mainly in low-count regions, which is mostly areas outside the head and low-affinity regions. While the former region is without much interest, the latter is often used as a reference. In this context a large part (e.g. cerebellum) is chosen and averaged to give a reference value. In this kind of studies it is crucial to have unbiased estimates of the reference region.

Nevertheless, the negative counts represents an inaccuracy it would be best to avoid. If areas are known to be zero (e.g. because they are outside the head), then the reconstruction should be restricted in this way. We shall return to this in sec. 5.6.

## Statistical analysis

Not all uses of the reconstructed images involves human interpretation. Currently, most statistical analysis of of emission tomographic imaging is based on the theory of *Gaussian Random Fields*.

The theory was first developed by Keith John Worsley et al. in [82], although this should be regarded as a precursor for [83] where the theory was solidified. Those equipped with the mathematical skills necessary to read this paper are encouraged to read it.

---

<sup>2</sup>One “unreasonable” method that do meet this requirement would be an algorithm which reconstructed any data as a zero image.

While random field theory is well beyond the scope of this thesis, we may nevertheless note that the noise of the reconstructed image is modeled as white noise filtered by a Gaussian kernel. Thus, to produce images useful in statistical analysis, the reconstruction algorithm should approximate this model.

∫∫∫

### Image continuity

Discretization of the reconstructed image is a necessary prerequisite for a reconstruction algorithm. Theoretically, one could attempt to determine the position of the emission of each count acquired during the scanning process. However, since the counts are modeled as independent events, we are in reality approximating a probability distribution of the emission positions.

If we assume this distribution is in  $L^1$  and non-zero values are confined to the interior of the scanner it follows from the Riemann-Lebesgue lemma [30] that it is band-limited<sup>3</sup>. The only relevant functions this author can imagine without this property is a finite sum of Dirac functions.

Band-limited functions can be discretized, as established by the sampling theorem [54]. For a sum of Dirac-functions, it is probably acceptable to bin the emission points in pixels of finite size, especially if we can make the pixels arbitrarily small.

If we reconstruct a series of images with an increasing number of pixels, it is desirable that these converges towards a stable solution which estimates this distribution. In a nutshell, a reconstructed image of size  $1024 \times 1024$  should be no worse than an image of size  $128 \times 128$ , if the underlying functions satisfies the requirements above.

As explained in sec. 4.6.1 the unmodified EM-algorithm relies on pixelation and does not have this property. In light of the discussion above, the method should always be regularized, e.g. using a sieve.

## 5.3 Statistical reconstruction

After considering the requirements, we now turn to the different strategies. The reconstruction literature is vast, and it is beyond the scope of this text to even attempt summarizing the existing work. For early methods, we refer to the references in sec. 4.9 p. 47.

We note, however, that the majority of work in the field in the past 10 years has centered around statistical methods. Those interested in these methods should consult the excellent review by Fessler [22], which treats many of these methods from an algorithmic point of view in a general setting of *surrogate functions*. All EM-algorithms, for example, may be viewed as an optimization algorithm using a surrogate function derived from a Kullback-Leibler distance [8].

Generally, these methods use an iterative scheme to maximize a probability that involves the scanner (described in chapter 3) and a prior. This has led some authors to characterize these algorithms under headings such as SIRA (Statistical

<sup>3</sup>Roughly,  $L^1$  means “absolutely integrable”, but also includes a requirement from measure theory; a rather difficult branch of mathematics. From a practical point of view, the last property excludes the Dirac function but can be ignored otherwise. The confinement property is a rephrasing of compact support.

Iterative Reconstruction Algorithms) or SIR. The following describes a few representative samples of this kind of information used by such methods.

Most algorithms use Bayesian inference at their core. A typical example of this approach was given by Bowsher et al. [11], who used a prior that encouraged regions of constant concentrations. They showed that this worked well on a phantom which matched this prior.

The approach was taken to an extreme by Philipsen et al. [57], who used adaptive snakes (an active contour model) in a weak-membranes model as a prior. The resulting objective function was optimized using a mean-field approximation in a simulated annealing algorithm. The author finds that papers like this should convince most people that the quest for improved images has sent researchers to areas very far from the algorithms currently used in medical imaging.

We also mention a paper by Lee et al. [48], who used a Bayesian method with a prior that encouraged piecewise linear areas. While the algorithms are quite typical, it differentiates itself by justifying the prior using autoradiography.

In a different approach, Ardekani et al. [3] used cross-entropy to reconstruct images using several MR-scans together with the emission scan. Most of the remaining literature builds a prior from the other modalities before the reconstructing begins.

A remarkable paper by Dougherty et al. [18] demonstrates how difficult it is to validate statistical reconstruction methods. The paper describes a phantom of a rat-brain, which was derived from tomographic scanings followed by autoradiography. This should result in a gold standard for a realistic image, but co-registering the emission data with the autoradiographic sections proved difficult.

## Discussion

Given the enormous body of work in statistical reconstruction, there has been remarkably little impact in medical usage. While this could be ascribed to inertia, there are factors that make statistical reconstruction unattractive in clinical use.

First, statistical methods rely on mathematical brain-models. This means that noisy measurements are guided towards meaningful brain-images, implying a tendency to turn noise into brain-like artifacts. This is inherent in the approach taken by such algorithms. We note that this is a common objection even for the EM-algorithm described in sec. 4.6 p. 42; arguably the simplest possible statistical reconstruction method.

Second, most statistical methods use statistical distributions in a hyper-space. It is very difficult to visualize this in a way that is meaningful to the user. Although this may be regarded as a “visualization problem”, it must be solved if statistical methods are to be practical.

Thirdly, the most advanced statistical methods are difficult to compare across patients. However, in clinical use it is common to compare an image with a set of normals, i.e. images produced from healthy volunteers.

Finally, the images produced by the statistical methods usually looks qualitatively different from the classical methods. As already argued in sec. 5.1, this does not bode well for clinical adoption.

Nevertheless, statistical methods have the ability to use very detailed models of the scanning process. If an appropriate prior can be build from experimental knowledge, and quantitative accuracy is the only concern, then statistical methods may well be the proper choice.

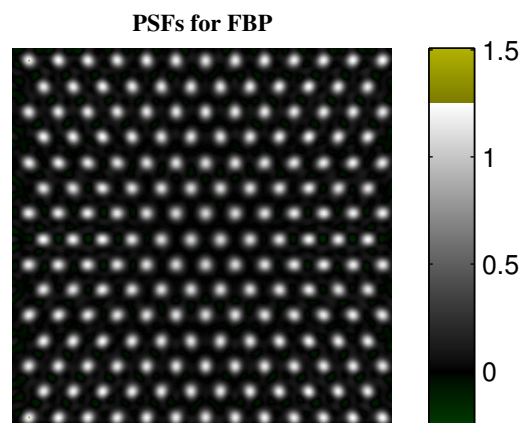
## 5.4 Linear reconstruction

∫∫∫

Filtered back-projection, as described in sec. 4.4 p. 38, is a linear reconstruction method. This means that the reconstructed image can be described in terms of a matrix:  $\hat{\mathbf{x}} = \mathbf{A}^\dagger \mathbf{b}$ . For FBP, this matrix was given by  $\mathbf{A}^\dagger = \mathbf{A}^T \mathbf{F}$ , where  $\mathbf{A}$  is the discrete Radon transform and  $\mathbf{F}$  is a filter, usually specified as a cut-off frequency and a window. Generally, the matrix  $\mathbf{A}^\dagger$  may originate from a number of algorithms.

One crucial property common to all linear methods, is that they are easy to characterize. Since they are linear, a given image will, on the average, be reconstructed as  $\mathcal{E}\{\mathbf{A}^\dagger \mathbf{b}\} = \mathbf{A}^\dagger \boldsymbol{\lambda} = \mathbf{A}^\dagger \mathbf{A} \mathbf{x}$ . Thus, if we let  $\mathbf{x}$  consists of a number of points, we can visualize the reconstructed PSFs. An example for the FBP is given in fig. 5.1.

**Figure 5.1:**  
Point-spread functions for the FBP-reconstruction using the parameters in fig. 4.7 on p. 42.



This leads to the question: how can this be improved without violating the user-requirements from sec. 5.1? Limiting new reconstruction methods to the exact same result as FBP does not constitute progress.

In the opinion of this author there is room for improvement in at least the following areas:

- **Make PSFs closer to homogeneous Gaussians:** It is worth noting from fig. 5.1 that the PSFs are elongated near the edges. Furthermore, the space between them contains negative values, even in this noiseless case. However, the PSFs are close enough to Gaussian that this should not make a qualitative difference in the look of the reconstructed images.
- **Increase resolution:** The whole reason for the project is to improve the resolution, which in the case of linear reconstruction is equivalent to make the PSFs more narrow. There would be no objections to this from the users, provided the PSFs remain near-Gaussian.

- **Incorporate anatomy:** Anatomical information should optionally be employed to get improved images. However, this is only likely to be successful if the user is in absolute control. The user must understand exactly what is going on, and be able to turn off any modification. This is particularly important in areas that are suspected to contain lesions and other irregularities.
- **Use more accurate system-matrices:** As described in chapter 3, the Radon transform is a crude scanner model, and incorporating effects such as scatter, attenuation, and beam-divergence is appropriate. This, however, is only meaningful if the user is allowed to switch them off.
- **Improve noise-models:** Only poor noise-models can be incorporated in strictly linear methods. However, quasi-linear methods may be applicable, where the reconstruction-matrix  $\mathbf{A}^\dagger$  is constructed using data. If necessary, the same matrix can be used to reconstruct any number of images, including normal material or different times in a dynamic series. Preferably, this model should still result in homogeneous Gaussian PSFs.

## Discussion

The most prominent example of a quasi-linear method is “Penalized Weighted Least Squares” by Fessler [20]. We note that this is one of the few methods that have been in routine use<sup>4</sup>.

Generally, linear methods satisfies the requirements in sec. 5.1 much better than statistical reconstruction methods. However, this comes at the prize of less accurate models:

- Concentrations may be negative.
- The discrete nature of the count-statistics can not be modeled.

This author believes that linear methods are easier to get adopted for clinical use than statistical methods, despite their shortcomings. Hence, the remaining part of the thesis will focus entirely on these methods. Chapter 6 deals with the issue of accuracy.

## 5.5 Regularized reconstruction

It should be evident from chapter 4 that any reconstruction method must suppress noise to create useful images. In general, this field is known as *regularization* and is important in many disciplines, e.g. seismology, astronomy, and signal processing to name a few [30].

As described above, the goal is to compute an approximate solution resulting in homogeneous Gaussian PSFs. In the literature, this is known as *mollifier methods*, and we can use an algorithm from chapter 4 to achieve this: first compute a solution using FBP ( $\omega_c = 1$  and no window) and apply a post-filter with a Gaussian kernel. If a more accurate solution is required, FBP may be used as a pre-conditioner in an iterative method (see p. 41).

<sup>4</sup>Adopted for routine use in cardiac scans at the University of Michigan Medical Center [21].

This offer only a small advantage over FBP, since the resolution is basically limited in the same way. However, the literature offers plenty of alternatives to post-filtering. Thus, we may use a post-filter (to imitate FBP), and increase the resolution but avoid excessive noise using regularization.

This appears like a nearly ideal way to deal with the requirements from the users. This is particularly true if the amount of post- and regularization-filter can be changed in real-time while the user is trying different settings. This is the method pursued in the remaining part of the thesis.

### Standard regularization

There are many standard regularization-schemes that can be expressed in terms of the *singular value decomposition* (SVD). The SVD of the system-matrix  $\mathbf{A}$  is given by:

$$(5.1) \quad \mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H$$

Here  $\mathbf{U}$  and  $\mathbf{V}$  have orthonormal columns, i.e.  $\mathbf{U}^H\mathbf{U} = \mathbf{V}^H\mathbf{V} = \mathbf{I}$  (this variant is known as the *thin SVD*), and  $\mathbf{\Sigma}$  is a square diagonal matrix with real, non-negative elements on the diagonal.

With this definition, the regularization schemes considered in this thesis may be written in terms of *filter-factors* [30]:

$$(5.2) \quad \mathbf{A}^\dagger = \sum_{i:\sigma_i>0} \frac{f_i}{\sigma_i} \mathbf{v}_i \mathbf{u}_i^H$$

Here  $f_i$  are the filter-factors, which can usually be specified as a function  $f_i = f(\sigma_i)$ . A few examples are given in tab. 5.1, although we shall not go further into the specific merits of these choices. The pseudo-inverse is as close to an inverse of  $\mathbf{A}$  as possible. All remaining filters share the property that  $f_i \approx 1$  for  $\sigma_i \gg \lambda$ , while  $f_i \approx 0$  for  $\sigma_i \ll \lambda$ . The parameter  $\lambda$  plays the same role as the cut-off frequency,  $\omega_c$ , did in the FBP studied in sec. 4.4 p. 38.

**Table 5.1:**  
Example of filter-factors studied in the literature [30].

Name	Filter-function
Pseudo-inverse	$f(\sigma) = 1$
Tikhonov	$f(\sigma) = \frac{\sigma^2}{\sigma^2 + \lambda^2}$
Truncated SVD (TSVD)	$f(\sigma) = \begin{cases} 1 & \text{for } \sigma \geq \lambda, \\ 0 & \text{otherwise.} \end{cases}$
Rutishauser	$f(\sigma) = \frac{\sigma^2}{\sigma^2 + \lambda^2 + \lambda^2/(\sigma^2 + \lambda^2)}$
Damped SVD	$f(\sigma) = \frac{\sigma}{\sigma + \lambda}$

### Tikhonov regularization

Apart from the pseudo-inverse, which is not ordinarily regarded as a form of regularization, by far the most studied method is Tikhonov regularization. It can be shown that the the Tikhonov solution,  $\hat{\mathbf{x}} = \mathbf{A}^\dagger \mathbf{b}$ , of the linear system  $\mathbf{A}\hat{\mathbf{x}} \approx \mathbf{b}$  is equivalent to the following optimization problem [30]:

$$(5.3) \quad \hat{\mathbf{x}} = \underset{\hat{\mathbf{x}}}{\operatorname{argmin}} \left( \|\mathbf{A}\hat{\mathbf{x}} - \mathbf{b}\|_2^2 + \lambda^2 \|\hat{\mathbf{x}}\|_2^2 \right)$$

Thus, the solution attempts to keep the residual error,  $\mathbf{A}\hat{\mathbf{x}} - \mathbf{b}$  small in norm, but adds a penalty to discourage solutions with large norms.

In a statistical setting, Tikhonov regularization is known as *ridge regression* [52]. If  $\mathbf{A}\mathbf{x} = \mathbf{b} + \mathbf{e}$ , where the elements in the noise-vector  $\mathbf{e}$  are uncorrelated random variables with zero mean but the same variance,  $s^2 < \infty$ , then there is a regularization parameter,  $\lambda$ , where the mean-square-error,  $\mathcal{E}\{\|\hat{\mathbf{x}} - \mathbf{x}\|_2^2\}$ , is less than the unregularized solution [43].

Tikhonov regularization may also be derived using a Bayesian approach. The regularized solution,  $\hat{\mathbf{x}}$ , is a maximum a posteriori estimate, assuming the elements of  $\mathbf{x}$  has expected zero mean and variance  $s^2/\lambda$  [43].

Finally, it can be shown that among the methods that can be defined in terms of filter-factors, the choice that minimizes  $\mathcal{E}\{\|\hat{\mathbf{x}} - \mathbf{x}\|_2^2\}$  is given by [30]:

$$(5.4) \quad f_i = \frac{\sigma_i^2}{\sigma_i^2 + s^2/(\mathbf{v}_i^T \mathbf{x})^2}$$

Since  $\mathbf{x}$  is unknown, we can only estimate the term in the denominator. Generally, using Tikhonov as an approximation is justified in tomography where the achievable resolution is low compared to the band-width of the object.

All derivations of Tikhonov regularization are based on the assumption that the elements in  $\mathbf{e}$  have the same variance. However, if the variance can be estimated, then we may instead use the *weighted least-squares* estimate:

$$(5.5) \quad \hat{\mathbf{x}} = \underset{\hat{\mathbf{x}}}{\operatorname{argmin}} \left( \|\mathbf{W}\mathbf{A}\hat{\mathbf{x}} - \mathbf{W}\mathbf{b}\|_2^2 + \lambda^2 \|\hat{\mathbf{x}}\|_2^2 \right)$$

Here  $\mathbf{W}$  is a diagonal matrix where the element  $\mathbf{W}_{(i,i)}$  is the inverse of the approximated standard deviation of  $\mathbf{e}_i$ . This function is the subject of chapter 6, and leads to a quasi-linear algorithm.

Another common variation of Tikhonov regularization is to change the penalty-term  $\|\hat{\mathbf{x}}\|_2$  to  $\|\mathbf{L}\hat{\mathbf{x}}\|_2$ , where  $\mathbf{L}$  is typically a discrete approximation of, say, the Laplace operator. However, in the current context this approach has several disadvantages:

- It introduces spatially correlated artifacts.
- The PSFs becomes dissimilar to Gaussian kernels.
- The form of  $\mathbf{L}$  must be justified from autoradiographic studies for each tracer, and such validation studies are extremely scarce.

For these reasons, we shall not use these methods, but instead rely on post-filtering.

∫∫∫

### Model mismatch

One convincing reason to use regularization methods based on filter-factors is their ability to reject model-mismatch. This stems from the fact that the scanner-model can be regarded as a perturbation of the true scanner, and we may therefore use results from perturbation-theory of the SVD to establish bounds on this error.



Several key results are, for other reasons, included in appendix C: Ostrowski's theorem (theorem C3 p. 147) states that multiplicative factors will at most change the singular values by the factors themselves. Thus, an inaccurate calibration of the photo-multiplier tubes or the absence of an attenuation model, will not change the singular values significantly.

Weyl's theorem (theorem C1 p. 147) takes care of the remaining errors, although somewhat less convincingly.

The reconstructed image depends not only on the singular values, but also singular vectors. A rigorous treatment depends on angles between subspaces, but the main result is that these are not severely affected by small perturbations. Thus, if we use a reconstruction method where the filter-factors do not change too rapidly, then we can expect that the reconstructed image is not detrimentally affected by the model mismatch.

We note that considerations of model-mismatch is very uncommon in the reconstruction literature [22].

## 5.6 Incorporating anatomy

There are two principal ways to use anatomy in reconstruction:

1. Build a prior, using probability- or information-theory, and apply a statistical reconstruction algorithm.
2. Impose constraints on the reconstructed image during the reconstruction.

As discussed in sec. 5.3, statistical methods relies on information that is difficult to convey accurately to non-technical users. This issue is made even more difficult when we add the requirement that the user must be able to control the process, including turning it off in different parts of the image.

This leads us to consider the second approach. We note that the idea of restricting the images was proposed at least as early as 1984 in one of the most cited papers on the EM-algorithm [45, p. 309]:

Certain equality constraints also become trivial to implement in the above framework.

...

Neighboring pixels can be forced to have the same intensities.

The paper also mentions the simplicity of incorporating areas known to be zero. One way to accomplish this is to set the initial iterate to zero, although this scheme must be modified if a convolutive sieve is used.

Using restrictions to incorporate anatomical knowledge has the attractive property that it is easy to visualize: a simple sketch of the constant regions suffice. It also has the advantage that many workflows used clinically already depends on identifying regions. Therefore the approach can be used together with familiar tools.

∫∫∫

### Anatomy and regularization

If we know that some pixels in  $\mathbf{x}$  are zero, then we may create an operator,  $\mathbf{P}$ , that zero out pixels known to be zero. Similarly, we may let  $\mathbf{P}$  be an operator that averages all pixels in one or more disjoint regions.

To be specific, let all pixels be categorized into  $K$  disjoint regions, many possibly containing a single pixel. Let  $\mathbf{p}_k$  be a vector with one element per pixel, and let an element be 1 if a pixel belonging to region  $k$  and zero otherwise. We may then write  $\mathbf{P}$  as follows:

$$(5.6) \quad \mathbf{P} = \sum_{k=1}^K (\mathbf{p}_k \mathbf{p}_k^T) / (\mathbf{p}_k^T \mathbf{p}_k)$$

It follows immediately that  $\mathbf{P} = \mathbf{P}^T$ . Since all regions are assumed to be disjoint we have  $\mathbf{p}_i^T \mathbf{p}_j = 0$  for  $i \neq j$ , and we get:

$$\begin{aligned} \mathbf{P}\mathbf{P} &= \left( \sum_{k_1=1}^K (\mathbf{p}_{k_1} \mathbf{p}_{k_1}^T) / (\mathbf{p}_{k_1}^T \mathbf{p}_{k_1}) \right) \left( \sum_{k_2=1}^K (\mathbf{p}_{k_2} \mathbf{p}_{k_2}^T) / (\mathbf{p}_{k_2}^T \mathbf{p}_{k_2}) \right) \\ &= \sum_{k_1=1}^K \sum_{k_2=1}^K (\mathbf{p}_{k_1} \mathbf{p}_{k_1}^T \mathbf{p}_{k_2} \mathbf{p}_{k_2}^T) / (\mathbf{p}_{k_1}^T \mathbf{p}_{k_1} \mathbf{p}_{k_2}^T \mathbf{p}_{k_2}) \\ &= \sum_{k=1}^K (\mathbf{p}_k \mathbf{p}_k^T \mathbf{p}_k \mathbf{p}_k^T) / (\mathbf{p}_k^T \mathbf{p}_k \mathbf{p}_k^T \mathbf{p}_k) \\ &= \sum_{k=1}^K (\mathbf{p}_k \mathbf{p}_k^T) / (\mathbf{p}_k^T \mathbf{p}_k) \\ &= \mathbf{P} \end{aligned}$$

Thus,  $\mathbf{P}$  is an *orthogonal projector* [73]. In any case we have  $\mathbf{x} = \mathbf{P}\mathbf{x}$ , since  $\mathbf{P}$  does not change an image that satisfies the specified restrictions. Thus we may compute a regularized solution to the following system:

$$(5.7) \quad \mathbf{A}\mathbf{P}\hat{\mathbf{x}} \approx \mathbf{b}$$

If we let  $\mathbf{A}\mathbf{P} = \mathbf{U}\Sigma\mathbf{V}^T$  be an SVD where only non-zero singular values are retained, then we have  $\hat{\mathbf{x}} \in \text{range}(\mathbf{V}) \subseteq \text{range}(\mathbf{P}^T) = \text{range}(\mathbf{P})$ . Thus, the solution produced by any regularized method described in sec. 5.5 will automatically satisfy the specified constraints.

## 5.7 References

The user-requirements in sec. 5.1 are rarely treated in the literature, except at commercial websites. As the author did not have access to their software, these websites can not be endorsed by giving references.

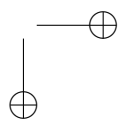
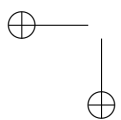
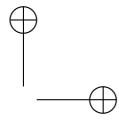
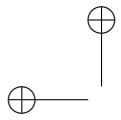
Anyone depending on the computational progress should be aware of the ITRS: International Technology Roadmap for Semiconductors [62, 63]. This organization produces high-quality reports with facts-based predictions.

Statistical reconstruction is a very active area of research. For a small sample we refer to the references given in sec. 5.3. Apart from pointing at a single review

[22], the best advice for those who wants to get into the field is to study the papers by Jeffrey A. Fessler.

Regularization of linear systems is standard material, which is covered in many places. However, in the setting of linear algebra (as opposed to more general Hilbert spaces), the place one should turn first is the doctoral thesis by Per Christian Hansen [30].

Projectors are described in many text-books on linear algebra, e.g. [71]. Another excellent reference is [73], which also use projectors in a numerical setting.



# Weighted least-squares

*Chapter 5 described a number of advantages of linear and quasi-linear reconstruction methods. However, one area where it lacked behind statistical methods was in quantitative accuracy. This chapter analyzes this behavior, and establish bounds on the difference between a simple weighted least-squares objective and a proper model of Poisson noise.*

*This study considers a broad class of reconstruction algorithms, and analyzes the worst-case behavior when a least-squares objective replaces a Poisson log-likelihood. We derive a bound on the Hausdorff-distance, in sinogram space, and compare it with statistical noise.*

*We show that in many circumstances a broad class of reconstruction algorithms can be expected to produce similar images using either objective. For rigor, we give a bound in the important special case where the penalty-function (or log-prior) is itself a weighted least-squares objective.*

**This chapter is mathematically oriented.**

## 6.1 Chapter overview

Most reconstruction algorithms can be stated as an optimization-problem of a function that contains a term that models the scanning process and a term that models an a priori knowledge or, equivalently, penalizes undesired images.

We consider a scanner that produces Poisson-distributed counts, and analyze the solutions when the Poisson log-likelihood is replaced with two different least-squares objectives. We bound the distance between solutions of equal likelihood and show that the reconstructed images should be similar in many circumstances. In the important special case, where both the log-likelihood and penalty-function (or log-prior) are weighted least-squares objectives, we give a rigorous bound.

Most studies evaluate a likelihood function as a part of a reconstruction process. The goal in this chapter is to isolate the effect of the likelihood function, and make as few assumptions as possible. To the best knowledge of the author, no previous work have established bounds that applies as generally as those presented here.

### Methods

First, the likelihood functions considered are introduced in sec. 6.2. The LS-error is seen as an approximation to the Poisson log-likelihood, and objectives for weighted LS (WLS) and model-weighted LS (MLS) schemes are derived in sec. 6.3 and 6.4.

Next, sec. 6.5 introduces a generalized view of regularized reconstruction algorithms. We consider iso-surfaces of the likelihood-functions, and define the Hausdorff distance between such surfaces. We show that small distances between such surfaces result in similar reconstructed images.

The bound obtained will in general depend on the level of regularization, and sec. 6.7 considers the statistical distribution of the Poisson log-likelihood. This leads to a range of practical threshold values for the log-likelihood.

Section 6.8 analyzes the WLS-scheme, and derives a bound on the distance between the likelihood surfaces. This bound is compared with the inherent statistical variation of a Poisson process.

Finally, sec. 6.9 analyzes the model-weighted LS scheme. Since the WLS-scheme is too simple to encompass regions with very few counts, the new relative bound is given. This demonstrates that the Poisson-likelihood and WLS-objective differ mainly by the quality of the variance estimates.

## 6.2 Likelihood functions

The acquisition process in emission tomography is inherently statistical, and is described in chapter 3. We shall assume that the system matrix  $\mathbf{A} \in \mathbb{R}^{M \times N}$  maps  $N$  pixels into  $M$  sinogram bins.

It is assumed the components of  $\mathbf{b}$  are independent and Poisson-distributed with parameters given by  $\boldsymbol{\lambda} = \mathbf{A}\mathbf{x}$ , where  $\mathbf{x}$  is the true object in the scanner. The likelihood of  $\boldsymbol{\lambda}$  can therefore be derived from the Poisson-distribution as described in sec. 3.8 p. 30:

$$(6.1) \quad L_P(\boldsymbol{\lambda}, \mathbf{b}) = \prod_{i=1}^M e^{-\lambda_i} \lambda_i^{b_i} / b_i!$$

We do not consider the likelihood functions directly. Since the projections in  $\mathbf{b}$  are independent stochastic variables, it is easier to work with the log-likelihood. The functions considered peak at  $\boldsymbol{\lambda} = \mathbf{b}$ , and we subtract this peak to get non-positive values.

The Gaussian likelihood resulting in least-squares problems is given by [52]:

$$L_{LS}(\boldsymbol{\lambda}, \mathbf{b}) = \frac{e^{-\frac{1}{2}(\mathbf{b}-\boldsymbol{\lambda})^T \boldsymbol{\Sigma}^{-1}(\mathbf{b}-\boldsymbol{\lambda})}}{\sqrt{\det(2\pi\boldsymbol{\Sigma})}}$$

where  $\boldsymbol{\Sigma} = \text{diag}([\sigma_1^2 \ \dots \ \sigma_M^2])$ .

Briefly postponing the estimation of  $\sigma_i^2$ , we may now define the log-likelihood functions. For convenience define  $0 \ln(0) = 0$  as a compound symbol and get:

$$(6.2) \quad \begin{aligned} \ell_P(\boldsymbol{\lambda}, \mathbf{b}) &= \ln(L_P(\boldsymbol{\lambda}, \mathbf{b})) - \ln(L_P(\mathbf{b}, \mathbf{b})) \\ &= \sum_{i=1}^M (b_i - \lambda_i + b_i \ln(\lambda_i) - b_i \ln(b_i)) \end{aligned}$$

$$(6.3) \quad \begin{aligned} \ell_{LS}(\boldsymbol{\lambda}, \mathbf{b}) &= \ln(L_{LS}(\boldsymbol{\lambda}, \mathbf{b})) - \ln(L_{LS}(\mathbf{b}, \mathbf{b})) \\ &= \sum_{i=1}^M -\frac{(\lambda_i - b_i)^2}{2\sigma_i^2} \end{aligned}$$

It will be necessary to distinguish between solutions associated with different likelihood functions. In this chapter, a hat decorates quantities that involve the Poisson-likelihood (e.g.  $\hat{\boldsymbol{\lambda}}$  and  $\hat{\mathbf{x}}$ ).

### 6.3 Weighted least-squares

If the variances are estimated directly from the measured sinogram, maximizing  $\ell_{\text{LS}}$  becomes a weighted least-squares (WLS) problem. Quantities will be decorated with an inverted hat, e.g.  $\hat{\lambda}$ .

In the simplest case, the approximation is restricted to estimate  $\sigma_i^2$  from a single projection, i.e.  $b_i$ . This represents a worst-case scenario with no smoothing in the sinogram, but have the advantage that no assumptions are required from  $\mathbf{A}$  or  $\mathbf{x}$ .

Expanding a single term of eq. (6.2) around  $\lambda_i = b_i$  gives:

$$(6.4) \quad \ell_{\text{P}}(\lambda_i, b_i) \approx -\frac{(\lambda_i - b_i)^2}{2b_i} + \frac{(\lambda_i - b_i)^3}{3b_i^2} - \frac{(\lambda_i - b_i)^4}{4b_i^3} + \dots$$

Choosing  $\sigma_i^2 = b_i$  makes the first term equal to eq. (6.3), but can not be used for  $b_i = 0$ . The analysis is based on the limit  $\sigma_i^2 \rightarrow 0^+$ , which effectively enforces  $\lambda_i = b_i = 0$ . Since  $\ell_{\text{P}}(0, 0) = 0$ , we also define  $\ell_{\text{WLS}}(0, 0) = 0$  and may now write the WLS-objective in the following form:

$$(6.5) \quad \ell_{\text{WLS}}(\boldsymbol{\lambda}, \mathbf{b}) = \sum_{\substack{i=1 \\ b_i \neq 0}}^M -\frac{(\lambda_i - b_i)^2}{2b_i}, \quad \lambda_i = 0 \text{ for } b_i = 0.$$

In practical methods, such as penalized weighted least-squares (PWLS),  $\sigma_i^2$  is estimated by filtering over the sinogram and considering detector efficiencies, dead-time correction, and transmission errors among other factors [20]. We therefore expect the analysis based on this function to be much worse than that encountered in practice, in particular for regions with very few counts.

### 6.4 Model-weighted least-squares

If the variances are allowed to depend on the estimated parameters, i.e.  $\sigma_i^2 = g(\lambda_i)$ , we get a model-weighted least-squares (MLS) problem. Quantities are decorated with a tilde, e.g.  $\tilde{\lambda}$ .

Although the MLS-problem is essentially as hard to optimize as the Poisson likelihood [22], it is of theoretical interest for several reasons. First, the analysis explains the behavior when a good approximation of the variance is available. Also, it helps to explain the behavior of the Poisson-likelihood in the region very near the measured sinogram where  $\tilde{\lambda} \approx \mathbf{b}$ .

Inserting  $\sigma_i^2 = g(\lambda_i)$  in eq. (6.3) and expanding yields:

$$\begin{aligned} \ell_{\text{MLS}}(\lambda_i, b_i) \approx & -\frac{(\lambda_i - b_i)^2}{2g(b_i)} + \frac{(\lambda_i - b_i)^3 g'(b_i)}{2g(b_i)^2} \\ & + \frac{(\lambda_i - b_i)^4}{4} \left( \frac{g''(b_i)}{g(b_i)^2} - 2 \frac{g'(b_i)^2}{g(b_i)^3} \right) + \dots \end{aligned}$$

Setting  $\sigma_i^2 = g(\lambda_i) = \lambda_i$  reproduces the first term in eq. (6.4), while the error in the second term is  $-(\lambda_i - b_i)^3 / (6b_i^2)$ , i.e. half that of  $\ell_{\text{WLS}}$  in absolute value. Thus,  $\ell_{\text{MLS}}$  is much better than  $\ell_{\text{WLS}}$  near the peak  $\lambda_i \approx b_i$  for this choice of  $\sigma_i^2$ .

For our purpose, with emphasis on low-count regions, a more interesting situation is the case  $\lambda_i \gg b_i$  and especially  $b_i = 0$ . Therefore we want the ratio  $\ell_{\text{MLS}}(\lambda_i, 0)/\ell_{\text{P}}(\lambda_i, 0) = \lambda_i/(2g(\lambda_i))$  to be 1. It follows that  $\sigma_i^2 = g(\lambda_i) = \lambda_i/2$ .

The two equations differ only by a constant factor. From an algorithmic point of view, this corresponds to different choices of the regularization parameter discussed in sec. 6.5. All further analysis will be based on the latter:

$$(6.6) \quad \ell_{\text{MLS}}(\boldsymbol{\lambda}, \mathbf{b}) = \sum_{i=1}^M -\frac{(\lambda_i - b_i)^2}{\lambda_i}$$

## 6.5 General reconstruction

As shown in chapter 4, the maximum-likelihood solution is rarely desirable, unless the spatial sampling is very crude. Instead, we search a smoother solution that strikes a balance between likelihood and other desirable features.

Most methods can be written in the form of an optimization problem.

**Definition 6.1: (Optimized reconstruction)** *Let  $\ell(\mathbf{A}\hat{\mathbf{x}}, \mathbf{b})$  be a log-likelihood function (e.g.  $\ell_{\text{P}}$  or  $\ell_{\text{WLS}}$ ) modeling the scanner acquisition, let  $r(\hat{\mathbf{x}})$  be a non-negative convex penalty-function, and let  $\alpha$  be a regularization constant.*

Furthermore, define the following function:

$$f(\hat{\mathbf{x}}, \mathbf{b}) = \ell_{\text{P}}(\mathbf{A}\hat{\mathbf{x}}, \mathbf{b}) - \alpha r(\hat{\mathbf{x}})$$

Then an optimizing reconstruction algorithm computes the following image over a convex set  $\mathcal{S}$ :

$$\hat{\mathbf{x}} = \underset{\hat{\mathbf{x}} \in \mathcal{S}}{\operatorname{argmax}} f(\hat{\mathbf{x}}, \mathbf{b})$$

The function  $r(\hat{\mathbf{x}})$  is either a penalty or log-prior depending on the philosophy and derivation of the method. Often the regularization parameter,  $\alpha$ , is considered separately. Although  $\alpha$  may be determined indirectly (e.g. by early stopping as described in sec. 4.7 p. 45), its function is to determine how large  $\ell_{\text{P}}(\mathbf{A}\hat{\mathbf{x}}, \mathbf{b})$  must be before a solution is acceptable.

Conversely, we may view the reconstruction as the process of maximizing the penalty-function over the iso-surface  $\ell_{\text{P}}(\mathbf{A}\hat{\mathbf{x}}, \mathbf{b}) = k$ . If  $\ell_{\text{LS}}$  approximates  $\ell_{\text{P}}$  well, then the solution  $\hat{\mathbf{x}}$  should have  $\ell_{\text{LS}}(\mathbf{A}\hat{\mathbf{x}}, \mathbf{b}) \approx k$ .

From an algorithmic point of view, fixing  $k$  instead of  $\alpha$  is often the goal. If  $\ell_{\text{WLS}}$  is combined with a weighted least-squares penalty-function, this strategy is known as the discrepancy principle [30].

A similar view was advocated by Veklerov and Llacer [76], who used the notion of *feasibility shell* to determine an appropriate sieve for the EM-algorithm. The shell corresponds to iso-surfaces for a small range of  $k$ , and the appropriate sieve is essentially equivalent to a fixed value of  $k$ .



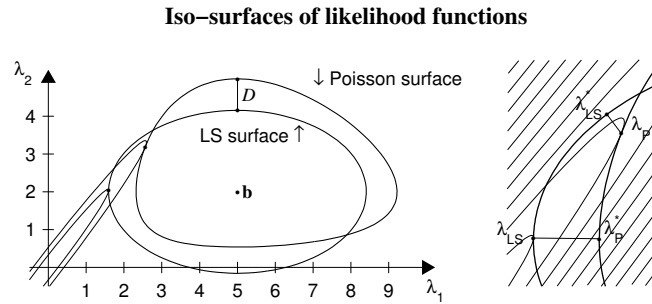
## 6.6 Hausdorff distance

To fix ideas we present a reduced example. Let the system matrix  $\mathbf{A}$  project two voxels ( $N = 2$ ) onto a two-element sinogram ( $M = 2$ ). Define the penalty-function as  $r(\mathbf{x}) = \|\mathbf{x}\|_2^2$  and assume we have measured the sinogram  $\mathbf{b}$ :

$$\mathbf{A} = \begin{bmatrix} 0.30 & 0.50 \\ 0.28 & 0.70 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$$

Fig. 6.1 shows iso-curves of the likelihood- and penalty functions. The log-likelihood is set at  $k = -0.58 \cdot 2$ , a choice discussed in sec. 6.7. The enlargement on the right shows the distance from each solution to the nearest point on the opposite surface.

**Figure 6.1:** Iso-surfaces of the likelihood- and the penalty-functions in sinogram-space, the maximum distance  $D$ , and solutions. The enlargement shows the solutions and their minimal displacement to the opposite surface.



Our approach will be to bound the distance between the surfaces in sinogram space. To make the statement precise, take  $\|\cdot\|$  to be an unspecified norm, and define the distance  $D$  as the maximum displacement from one surface to another:

$$\begin{aligned} D_{1 \rightarrow 2} &= \max_{\lambda_1 \in S_1} \min_{\lambda_2 \in S_2} \|\lambda_1 - \lambda_2\| \\ D_{2 \rightarrow 1} &= \max_{\lambda_2 \in S_2} \min_{\lambda_1 \in S_1} \|\lambda_1 - \lambda_2\| \\ (6.7) \quad D &= \max\{D_{1 \rightarrow 2}, D_{2 \rightarrow 1}\} \end{aligned}$$

The surfaces are the iso-surfaces of the likelihood-functions and depend on the threshold-value  $k$ . Although the displacements  $D_{1 \rightarrow 2}$  and  $D_{2 \rightarrow 1}$  were identical in the example, it is generally necessary to consider them separately.

The distance  $D$  is called the Hausdorff-distance between  $S_1$  and  $S_2$  with respect to the given norm [1]. It is named after Felix Hausdorff, who showed that  $D$  is a distance, and  $D_{1 \rightarrow 2}$  and  $D_{2 \rightarrow 1}$  are metrics.  $D$  is sometimes known as the “undirected Hausdorff distance”.

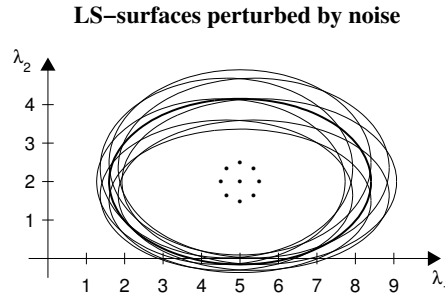
### Implications in image-space

Bounds on  $D$  are useful because they have consequences in image-space. The distance between the solutions in fig. 6.1 are larger than  $D$  only because the penalty differ less between the solutions than the opposite points. This must be the case since, by definition, the solutions found on each surface have the lowest penalty. In other words, the LS-solution is better (measured by the penalty-function) than the Poisson-solution perturbed at most  $D$  in the sinogram.

Consider the difference, in image-space, between the solutions above measured in the 2-norm, i.e. the penalty function. The transformation (the inverse of  $\mathbf{A}$ ) squeezes the penalty-curves into circles. Replacing the Poisson-surface with an LS-surface caused a smaller change in the penalty-function than any of the displacements.

To emphasize this point, fig. 6.2 shows LS-surfaces that arise from small perturbations of  $\mathbf{b}$ . All these surfaces must result in similar images in image-space, because they represent the same object perturbed by statistical noise. It should be evident that in general, any surface embedded in the “cloud” should result in similar images.

**Figure 6.2:**  
Cloud of LS-surfaces resulting from perturbing  $\mathbf{b}$ .



In the important special case where the penalty-function is on the form  $r(\mathbf{x}) = \|\mathbf{L}\mathbf{x}\|_2^2$ , theorem A2 in appendix A shows that under mild assumptions there are images satisfying the relation  $r(\hat{\mathbf{x}} - \tilde{\mathbf{x}}_1) + r(\hat{\mathbf{x}} - \tilde{\mathbf{x}}_2) \leq r(\tilde{\mathbf{x}}_2) - r(\tilde{\mathbf{x}}_1)$  for some WLS-reconstructed images,  $\tilde{\mathbf{x}}_1$  and  $\tilde{\mathbf{x}}_2$ .

These solutions must, by definition, minimize the penalty-function on some surfaces in the cloud. If we can guarantee that all surfaces result in similar images, by bounding  $D$ , then it follows that the reconstructed image is near the image reconstructed using the Poisson surface.

In the general case this result must be weakened. Section A2 outlines a proof that any vector  $\mathbf{d}_\lambda$  have a  $\mathbf{d}_b$  so that  $\ell_{\text{WLS}}(\hat{\lambda} + \mathbf{d}_\lambda, \mathbf{b}) = \ell_{\text{WLS}}(\hat{\lambda}, \mathbf{b} - \mathbf{d}_b)$  where  $(d_b)_i = c_i(d_\lambda)_i, |c_i| < 1 + \sqrt{2} < 2.42$ . This ensures that if  $D$  is small, compared to the statistical noise of  $\mathbf{b}$ , the Poisson-surface will be embedded in the cloud. Thus, the images reconstructed using the Poisson-surface and the WLS-surface can only vary greatly, if the images reconstructed using the Poisson-surface is sensitive to statistical noise.

### Non-negativity

The distance bounds apply unmodified for algorithms that enforce non-negativity in the image-space. To see this, observe that such an algorithm must be able to choose a solution on the boundary of the non-negative set.

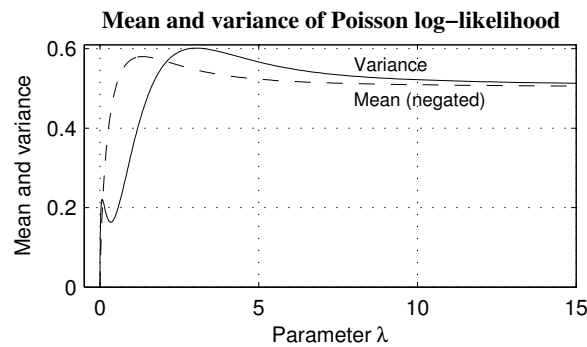
Since the non-negative vectors in image-space form a convex set, so do the set in sinogram-space after a linear transformation. The iso-surfaces treated in this section are therefore cut so they lie inside this convex set. However, the distance between two surfaces modified in this way can never exceed that of the original surfaces. If the original opposite point has been removed, then there must be a closer one lying on the surface of the non-negative set.

## 6.7 Log-likelihood range

Before comparing the likelihood functions we will start by determining the range of likelihood-thresholds that is useful in practice. To this end, consider the distribution of  $\ell_P(\boldsymbol{\lambda}, \mathbf{b})$  as a stochastic variable when  $\boldsymbol{\lambda}$  is fixed.

Since the total log-likelihood  $\ell_P(\boldsymbol{\lambda}, \mathbf{b}) = \sum_{i=1}^M \ell_P(\lambda_i, b_i)$  is a sum of many independent terms, it should have a nearly Gaussian distribution. Fig. 6.3 shows the negated mean and variance for a single term.

**Figure 6.3:** Mean and variance of the Poisson log-likelihood as a function of the true parameter  $\lambda_i$ . The mean is negated for illustrative convenience. Both are bounded, implying a limited range of useful values of the log-likelihood.



### Mean and variance

By definition, the mean log-likelihood of a projection is:

$$\begin{aligned} \mathcal{E}\{\ell_P(\lambda_i, b_i)\} &= \sum_{b_i=0}^{\infty} \frac{e^{-\lambda_i} \lambda_i^{b_i}}{b_i!} \ell_P(\lambda_i, b_i) \\ &= \sum_{b_i=0}^{\infty} \frac{e^{-\lambda_i} \lambda_i^{b_i}}{b_i!} (\lambda_i \ln(\lambda_i) - b_i \ln(b_i)) \end{aligned}$$

One important point is that  $\mathcal{E}\{\ell_P(\lambda_i, b_i)\} \geq -0.5802$ . This gives a lower bound of the mean, *irrespective* of the unknown parameter  $\lambda_i$ . Similarly, the variance is bounded above by  $\text{var}(\ell_P(\lambda_i, b_i)) < 0.6015$  and asymptotically becomes  $1/2$  for  $\lambda_i \rightarrow \infty$ .

### Likelihood threshold

As mentioned we expect that  $\ell_P(\boldsymbol{\lambda}, \mathbf{b})$  has a nearly Gaussian distribution. Combining this with the result above, it follows that the threshold  $k = -0.5802M$  should give  $\ell_P(\boldsymbol{\lambda}, \mathbf{b}) > k$  at least 50% of the time for the true  $\boldsymbol{\lambda}$ .

Setting  $k = -0.5802M(1 + \epsilon)$  adds a confidence interval, where  $\epsilon$  represents a few standard deviations. Since the variance was bounded, it follows that  $\epsilon$  decreases as  $1/\sqrt{M}$ . Usually  $M$  is sufficiently large that  $\epsilon$  can safely be ignored.

For  $k$  much lower than  $-0.5802M$ , we accept solutions,  $\hat{\mathbf{x}}$ , significantly less likely to produce the measured sinogram,  $\mathbf{b}$ , than the true concentration  $\mathbf{x}$ . Usually  $k$  should be higher, especially if there are many low-count projections ( $\lambda_i < 1$ ) as is often the case for real scans.

## 6.8 Analysis of weighted least-squares

Let  $\hat{\lambda}$  and  $\check{\lambda}$  be regularized solutions obtained using the Poisson likelihood and the WLS-likelihood respectively. These are called primary estimates.

We now have to find a displacement vector from each primary estimate to the opposite surface (see fig. 6.1 in sec. 6.5.) We want to bound a norm of these displacement vectors.

Consider replacing each component with a new value that matches the likelihood of the opposite surface. The displacement always overestimates the distance defined in eq. (6.7).

For instance, let  $S_1 = \{\hat{\lambda} | \ell_P(\hat{\lambda}, \mathbf{b}) = k\}$  and  $S_2 = \{\check{\lambda} | \ell_{WLS}(\check{\lambda}, \mathbf{b}) = k\}$ . Now replace each component of  $\hat{\lambda}$  with a suitable entry so  $\check{\lambda}^*$  satisfies  $\ell_P(\check{\lambda}^*, \mathbf{b}) = \ell_{WLS}(\check{\lambda}^*, \mathbf{b})$ . Finally, an upper bound on  $\|\hat{\lambda} - \check{\lambda}^*\|$  must also be an upper bound on  $D_{1 \rightarrow 2}$ .

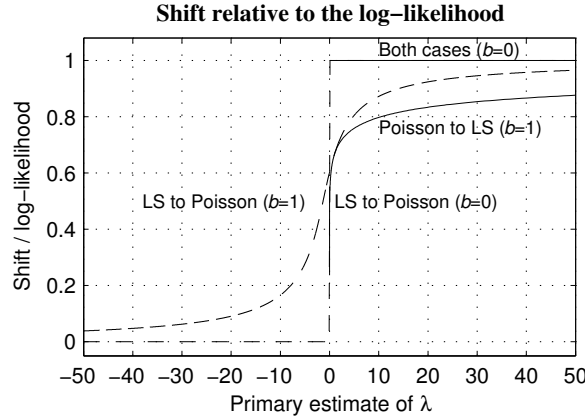
### Surface distance bound

First, assume that the primary Poisson-estimate is given. Equating terms in eq. (6.3) and eq. (6.5) and solving for the parameter yields:

$$(6.8) \quad \check{\lambda}_i^* = b_i + \text{sgn}(\hat{\lambda}_i - b_i) \sqrt{-2b_i \ell_P(\hat{\lambda}_i, b_i)}$$

Plotting  $\Delta_i / \ell_P(\hat{\lambda}_i, b_i)$ , where  $\Delta_i = \check{\lambda}_i^* - \hat{\lambda}_i$ , reveals an important property of the displacement:  $\ell_P(\hat{\lambda}_i, b_i) \leq \Delta_i \leq 0$ . Fig. 6.4 shows only  $b_i = 0$  and  $b_i = 1$ , since for  $b_i \neq 1$  the displacement is a function of  $\lambda_i / b_i$ . Thus, for  $b_i \geq 1$  one should read the  $\lambda$ -axis as  $\lambda_i / b_i$  and use the curve for  $b_i = 1$ .

**Figure 6.4:** Relative displacement for equating a single term of the likelihood-functions. **Solid:** primary estimate on the Poisson-surface (undefined for  $\lambda_i \leq 0$ ). **Dashed:** relative displacement from a primary estimate on the LS-surface.



Using the observation above and summing over all projections immediately gives the bound:

$$\|\check{\lambda}^* - \hat{\lambda}\|_1 \leq -\ell_P(\hat{\lambda}, \mathbf{b}) = -\ell_{WLS}(\check{\lambda}^*, \mathbf{b})$$

Analogously, equating terms in eq. (6.3) and eq. (6.6) bounds the displacement from a point on the LS-surface. The dashed curve in fig. 6.4 shows the result, but it is necessary to solve for  $\hat{\lambda}_i^*$  numerically. The same argument as before applies, and we obtain:

$$\|\check{\lambda} - \hat{\lambda}^*\|_1 \leq -\ell_P(\hat{\lambda}^*, \mathbf{b}) = -\ell_{WLS}(\check{\lambda}, \mathbf{b})$$

It is convenient to combine the equations into a single equation, without specifying the primary surface:

$$(6.9) \quad \|\tilde{\lambda} - \hat{\lambda}\|_1 \leq D \leq -\ell_P(\hat{\lambda}, \mathbf{b}) = -\ell_{\text{WLS}}(\tilde{\lambda}, \mathbf{b}) = -k$$

### Comparison with noise

Whether the bound given in eq. (6.9) is acceptable depends on the norm of  $\mathbf{b}$  and the remaining errors. Since  $\|\mathbf{b}\|_1$  is the total number of counts, the result in sec. 6.7 suggests that the error is negligible if  $\|\mathbf{b}\|_1 \gg 0.58M$ , where  $M$  again represents the number of projections. However, to be quantitative we can compare the bound on  $D$  with the statistical fluctuation  $\mathcal{E}\{\|\mathbf{b} - \boldsymbol{\lambda}\|_1\}$ , where  $\boldsymbol{\lambda} = \mathbf{A}\mathbf{x}$  is a true (unknown) fixed parameter vector.

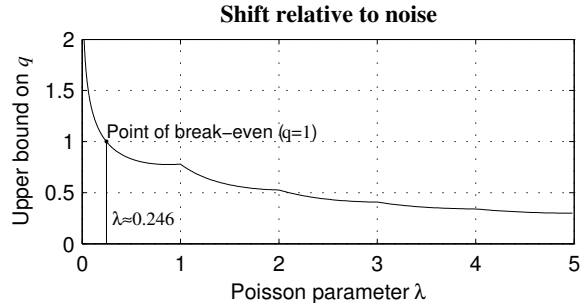
The fluctuation can be arbitrarily small when  $\boldsymbol{\lambda} \approx \mathbf{0}$ , but then the log-likelihood threshold should also be close to zero. We assume that the reconstruction is consistent in the sense that  $\ell_P(\hat{\lambda}, \mathbf{b}) \geq \mathcal{E}\{\ell_P(\boldsymbol{\lambda}, \mathbf{b})\} (1 + \epsilon)$ , where the last expectation is over  $\mathbf{b}$  and  $\epsilon$  is small (see sec. 6.7.) It now follows from eq. (6.9):

$$(6.10) \quad q = \frac{\|\tilde{\lambda} - \hat{\lambda}\|_1}{\mathcal{E}\{\|\mathbf{b} - \boldsymbol{\lambda}\|_1\}} \leq \frac{-\ell_P(\hat{\lambda}, \mathbf{b})}{\mathcal{E}\{\|\mathbf{b} - \boldsymbol{\lambda}\|_1\}} \leq \frac{(1 + \epsilon) \sum_{i=1}^M \mathcal{E}\{-\ell_P(\lambda_i, b_i)\}}{\sum_{i=1}^M \mathcal{E}\{|b_i - \lambda_i|\}}$$

The worst case occurs for  $\lambda_i = \|\mathbf{b}\|_1/M$ , shown in fig. 6.5, which gives:

$$(6.11) \quad q \leq \mathcal{E}\{-\ell_P(\lambda_i, b_i)\} / \mathcal{E}\{|b_i - \lambda_i|\}$$

**Figure 6.5:**  
Ratio between displacement-bound and statistical variation for the worst-case scenario  $\lambda_i = \|\mathbf{b}\|_1/M$ .

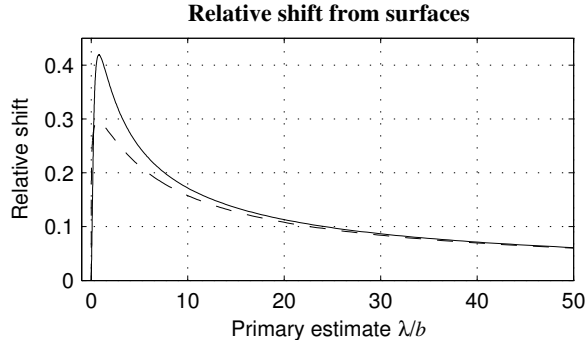


The displacement becomes larger than the fluctuation when  $\lambda_i$  is very small. Nevertheless, if  $\lambda_i$  is large for some  $i$ , the terms associated with very small  $\lambda_i$  have little effect on the fraction in eq. (6.10). Break-even happens around  $\lambda_i \approx 0.2464$ . We conclude that unless the majority of projections have an expected value less than 0.25, then the perturbation of the surface made by replacing the Poisson likelihood with the WLS-likelihood is no larger than what should be expected from statistical noise.

## 6.9 Analysis of model-weighted least-squares

The MLS-objective, defined by eq. (6.6), uses an improved estimate of the variance. This function equals the Poisson-function for  $b_i = 0$ , and it suffices to study  $b_i > 0$ .

**Figure 6.6:** Relative displacement between surfaces for  $b_i > 0$  where the dashed curve is from LS to Poisson, i.e.  $\tilde{s}_i$ , and the solid curve is Poisson to LS, i.e.  $\hat{s}_i$ . The relative displacement is less than 0.42.



Consider the relative displacements  $\hat{s}_i = (\hat{\lambda}_i - \tilde{\lambda}_i)/(\hat{\lambda}_i - b_i)$  and  $\tilde{s}_i = (\tilde{\lambda}_i - \hat{\lambda}_i)/(\tilde{\lambda}_i - b_i)$ . Fig. 6.6 shows both  $\hat{s}_i$  and  $\tilde{s}_i$  as functions of  $\lambda_i/b_i$ , and again the displacements are bounded.

The observation from fig. 6.6 leads to another bound. Writing  $\tilde{\lambda}^* = \hat{\lambda} + \mathbf{e}$  or  $\hat{\lambda}^* = \tilde{\lambda} - \mathbf{e}$  we have  $0 \leq e_i < 0.42b_i$ . Thus,  $\|\mathbf{e}\| = \|\tilde{\lambda} - \hat{\lambda}\| < 0.42\|\hat{\lambda} - \mathbf{b}\|$  measured in any norm.

We conclude that the iso-surfaces are relatively close, in the sense that if the Poisson-surface is close to the measured sinogram, then the distance to the MLS-surface is not much further away. This result holds for any choice of likelihood threshold and any subset of projections.

## 6.10 Summary of the analysis

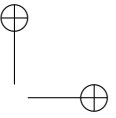
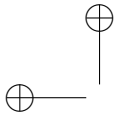
This chapter analyzed the consequences of replacing a Poisson-likelihood with a Gaussian likelihood. Two simple least-squares objectives were defined as approximations to the Poisson log-likelihood.

We bounded the Hausdorff-distance between the Poisson- and LS-likelihood surfaces in sinogram-space. It was shown that exchanging the Poisson-likelihood with a WLS-objective perturbed the surface less than the expected statistical noise in regions with more than about 0.25 average counts per projection. A model-weighted LS-objective always generated solutions with a bounded relative distance between the Poisson-solution.

We argued that perturbing the likelihood-surfaces less than the expected statistical noise implies that the reconstructed images are similar in image-space. This was proved rigorously in the case of Tikhonov-regularization.

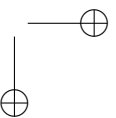
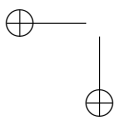
The analysis derived loose upper bounds of a worst-case analysis of a greatly simplified scenario. No statistical cancellation was introduced to improve the least-squares errors, and the weighted LS-scheme was restricted to estimating the variance from a single projection. It is surprising that a scheme that simple can achieve any degree of accuracy in areas with fractional average counts.

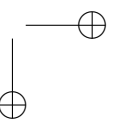
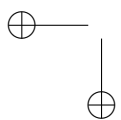
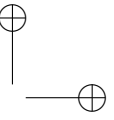
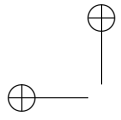
Even though no proof can be given that applies generally to all penalty-functions, there is ample reason to believe that weighted LS-objectives form very good approximations to the Poisson log-likelihood under practical circumstances. The model-weighted least-squares objective demonstrated that the main reason the



Poisson-likelihood do better in regions with very few counts is the ability to estimate fractional variances more than the particular form of the likelihood.

An important consequence is that there is sound theoretical justification for applying the tools in linear algebra to the problem of reconstructing Poisson-sources.



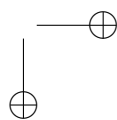
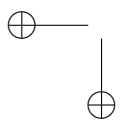
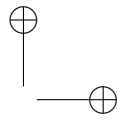
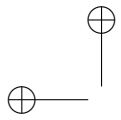






**Part III**

**Numerical Algorithms**



# Iterative framework

*This chapter develops a framework for computing the singular value decomposition. The algorithms are based on matrix-vector products, and operates on parts of the SVD, i.e. without requiring a full SVD to reside in memory at a single time.*

*The framework consists of a number of building-blocks, or modules. To put this in context, we show how several traditional algorithms can be expressed in terms of these modules. For generality, we allow any operator to be complex.*

**All chapters in this part are mathematically or algorithmically oriented.**

## 7.1 Partial SVD

The goal of this chapter is to develop modules for approximating the SVD of a matrix,  $\mathbf{A}$ , without forming  $\mathbf{A}$  explicitly. We will only access the matrix through matrix-vector multiplications by  $\mathbf{A}$  and  $\mathbf{A}^H$ , since these operations are often the only practical operations in large-scale problems.

In large-scale problems the SVD must be defined without needing all vectors simultaneously. One way to do this is to define a *singular triplet*.

**Definition 7.1: (Singular triplet)** *A singular triplet  $(\mathbf{u}_i, \mathbf{v}_i, \sigma_i)$  of a matrix,  $\mathbf{A} \in \mathbb{C}^{m \times n}$ , consists of two unit vectors,  $\mathbf{u}_i \in \mathbb{C}^{m \times 1}$  and  $\mathbf{v}_i \in \mathbb{C}^{n \times 1}$ , and a scalar,  $\sigma_i$ , which satisfies the equations:*

$$(7.1a) \quad \mathbf{A}\mathbf{v}_i = \sigma_i\mathbf{u}_i$$

$$(7.1b) \quad \mathbf{A}^H\mathbf{u}_i = \sigma_i^*\mathbf{v}_i$$

For brevity the term “singular” may be dropped when the context is clear. An *exact (singular) triplet* satisfies these relations exactly. In contrast a *numerical (singular) triplet* only satisfies  $\mathbf{A}\mathbf{v}_i \approx \sigma_i\mathbf{u}_i$  and  $\mathbf{A}^H\mathbf{u}_i \approx \sigma_i^*\mathbf{v}_i$ . We refer to sec. C6 p. 151 in the appendix for more details about singular triplets.

We will say that two singular triplets are *mutually orthogonal* if their singular vectors are orthogonal, i.e.  $\mathbf{v}_n^H\mathbf{v}_m = \mathbf{u}_n^H\mathbf{u}_m = 0$  for  $n \neq m$ . Again the term “mutual” may be dropped, and *numerically orthogonal* triplets only have approximately orthogonal singular vectors.

It is often advantageous to work on a number of singular triplets at a time. This is most conveniently done by collecting the singular triplets in matrices:

$$\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots] \quad \mathbf{B} = \text{diag}(\sigma_1, \sigma_2, \dots) \quad \mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \dots]$$

We shall refer to a collection of orthogonal singular triplets as a *partial SVD* (PSVD). For reference, we give a formal definition of the partial SVD.

**Definition 7.2: (Partial SVD)** A size- $k$  PSVD of a matrix  $\mathbf{A} \in \mathbb{C}^{m \times n}$  is a triplet of matrices,  $(\mathbf{U}, \mathbf{V}, \mathbf{B})$ , where  $\mathbf{U} \in \mathbb{C}^{m \times k}$  and  $\mathbf{V} \in \mathbb{C}^{n \times k}$ . Also  $\mathbf{U}^H \mathbf{U} = \mathbf{I}$  and  $\mathbf{V}^H \mathbf{V} = \mathbf{I}$  and finally the following equations must be satisfied:

$$(7.2a) \quad \mathbf{A}\mathbf{V} = \mathbf{U}\mathbf{B}$$

$$(7.2b) \quad \mathbf{A}^H \mathbf{U} = \mathbf{V}\mathbf{B}^H$$

### Numerical Partial SVD (NPSVD)

A PSVD is either exact or numeric, depending on whether the triplets are exact or numeric. Exact triplets with distinct singular values are always orthogonal (see theorem C14 on p. 152).

Unlike exact triplets, numerical triplets are not automatically mutually orthogonal, and it is necessary to include an error-term in the form of a residual. This leads to the Numerical Partial SVD (NPSVD), which is defined in the box.

To keep the correspondence between individual triplets the matrix  $\hat{\mathbf{B}}$  must be square. Note that some texts include part of the residual into  $\hat{\mathbf{U}}$  or  $\hat{\mathbf{V}}$  instead.

#### Numerical Partial SVD (NPSVD)

A size- $k$  NPSVD of a matrix  $\mathbf{A} \in \mathbb{C}^{m \times n}$  is a triplet of matrices,  $(\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}})$ , and residuals,  $\mathbf{E}_U \in \mathbb{C}^{m \times k}$  and  $\mathbf{E}_V \in \mathbb{C}^{n \times k}$ , where  $\hat{\mathbf{U}} \in \mathbb{C}^{m \times k}$ ,  $\hat{\mathbf{V}} \in \mathbb{C}^{n \times k}$ , and  $\hat{\mathbf{B}} \in \mathbb{C}^{k \times k}$ .

The matrices are related by the defining equations:

$$(7.3a) \quad \mathbf{A}\hat{\mathbf{V}} \triangleq \hat{\mathbf{U}}\hat{\mathbf{B}} + \mathbf{E}_U$$

$$(7.3b) \quad \mathbf{A}^H \hat{\mathbf{U}} \triangleq \hat{\mathbf{V}}\hat{\mathbf{B}}^H + \mathbf{E}_V$$

There must be a known constant,  $\epsilon_o < 1$ , such that the columns of  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  satisfies  $1 - \epsilon_o \leq \|\hat{\mathbf{u}}_i\|_2 \leq 1 + \epsilon_o$  and  $1 - \epsilon_o \leq \|\hat{\mathbf{v}}_i\|_2 \leq 1 + \epsilon_o$ .

If  $\hat{\mathbf{B}}$  is diagonal and  $\epsilon_o \ll 1$ , then the columns of  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  approximates the left and right singular vectors, while the entries in  $\hat{\mathbf{B}}$  approximates the singular values.

Ideally,  $\hat{\mathbf{U}}^H \hat{\mathbf{U}} = \mathbf{I}$ ,  $\hat{\mathbf{V}}^H \hat{\mathbf{V}} = \mathbf{I}$ , and  $\hat{\mathbf{B}}$  consist of a diagonal of real, non-negative entries. In this case the columns of  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  approximates left and right singular vectors and the diagonal of  $\hat{\mathbf{B}}$  approximates the singular values. The quality of these approximations depends on the residual norms,  $\|\mathbf{E}_U\|$  and  $\|\mathbf{E}_V\|$ . If the norms are zero the triplets are exact.

For the purpose of numerical software, we will not require that  $\hat{\mathbf{B}}$  is diagonal (this is why we use  $\hat{\mathbf{B}}$  instead of  $\hat{\mathbf{\Sigma}}$ ). It will be shown in sec. 7.2 that we may diagonalize  $\hat{\mathbf{B}}$  when needed.

As an aid to explain the theoretical background of the algorithms, it is helpful to name the quantities involved in an NPSVD. We shall use the terms shown in table 7.1. For brevity the *residual matrices* may be referred to as residuals if the context is clear.

### Numerical considerations

**Table 7.1:**  
Terminology used  
in the descriptions.

Quantity	Term
$\hat{\mathbf{B}}$	Coupling matrix
$\hat{\mathbf{U}}, \hat{\mathbf{V}}$	Left and right singular matrices
$\mathbf{E}_{\mathbf{U}}, \mathbf{E}_{\mathbf{V}}$	Left and right residual matrices

The goal is to develop numerically robust software that improves an NPSVD iteratively in a way that drives the residual norms to zero. Just as it is preferable to allow the coupling-matrix to be on a non-diagonal form, it is also preferable not to require that  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  have orthogonal columns.

In spite of these relaxations, it is convenient to think of the NPSVD as an approximation of a partial SVD. However, we will be careful to note when a derived quantity relies on a particular property of the singular matrices or the coupling matrix.

The NPSVD is given as a definition where  $\hat{\mathbf{U}}, \hat{\mathbf{V}},$  and  $\hat{\mathbf{B}}$  usually represents computed quantities (and are decorated with a hat). The residuals,  $\mathbf{E}_{\mathbf{U}}$  and  $\mathbf{E}_{\mathbf{V}}$ , are known only through eq. 7.3 and absorbs the rounding errors.

The chapter derives a number of modules that transform an NPSVD. We use the term module, because “update” (and its analogue sometimes known as “down-date”) have traditional meanings. Notationally, we will decorate quantities of a transformed NPSVD with tildes, i.e.  $\tilde{\mathbf{U}}, \tilde{\mathbf{V}}, \tilde{\mathbf{B}}, \tilde{\mathbf{E}}_{\mathbf{U}},$  and  $\tilde{\mathbf{E}}_{\mathbf{V}}$ .

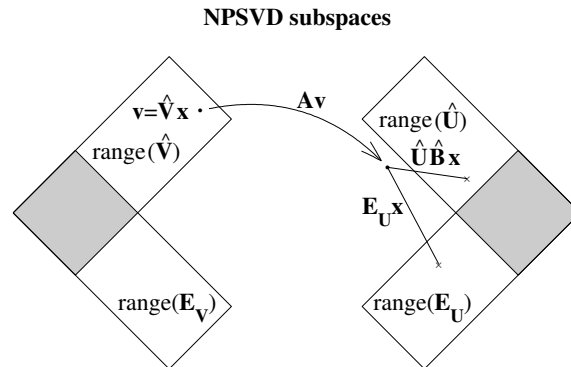
## Interpretation

It is helpful to have an intuitive understanding of the NPSVD, which may be attained by considering a specific problem. In scanner terminology  $\text{range}(\hat{\mathbf{U}})$  represents a space of sinograms and  $\text{range}(\hat{\mathbf{V}})$  represents a space of objects, i.e. brains in neurology.

When an object is put into the scanner, its emission concentration may be approximated by a vector in  $\mathbf{v} \in \text{range}(\hat{\mathbf{V}})$ . This (approximated) object may then be projected to the sinogram as the matrix-vector product  $\mathbf{u} = \mathbf{A}\mathbf{v}$ .

If the residuals in the NPSVD are known, then this multiplication which projects  $\mathbf{v}$  into the sinogram  $\mathbf{u}$ , can be done using the NPSVD as illustrated<sup>1</sup> in fig. 7.1.

**Figure 7.1:**  
Forward projection  
of a vector. In  
general  $\text{range}(\hat{\mathbf{U}})$   
and  $\text{range}(\mathbf{E}_{\mathbf{U}})$   
may overlap. After  
the projection the  
component lying  
in  $\text{range}(\mathbf{E}_{\mathbf{U}})$   
prevent a backpro-  
jection.



<sup>1</sup>This figure was inspired by a similar figure in [71].

We can also take a sinogram  $\mathbf{u} \in \text{range}(\hat{\mathbf{U}})$  and compute  $\mathbf{A}^H \mathbf{u}$ , i.e. back-project the sinogram into object space. However, after a projection, or back-projection, the residuals add a component that does not lie in the range of  $\hat{\mathbf{U}}$  (or  $\hat{\mathbf{V}}$ ), so we can not project again.

If the residual norms are negligible, we can also use the NPSVD to determine the object that projects to a given sinogram, i.e. solve the linear system. Although the NPSVD can be used for this purpose without decomposing it into singular triplets, it is nevertheless a useful way to think about the NPSVD.

## 7.2 Diagonalizing the NPSVD

Eventually we want the coupling-matrix  $\hat{\mathbf{B}}$  to be diagonal. Fortunately,  $\hat{\mathbf{B}}$  is generally of small dimension (compared to  $\mathbf{A}$ ), and it is therefore possible to compute the SVD of  $\hat{\mathbf{B}} = \mathbf{U}_B \boldsymbol{\Sigma}_B \mathbf{V}_B^H$ . Inserting this into the NPSVD in eq. (7.3) gives:

$$(7.4a) \quad \mathbf{A}(\hat{\mathbf{V}}\mathbf{V}_B) = (\hat{\mathbf{U}}\mathbf{U}_B)\boldsymbol{\Sigma}_B + \mathbf{E}_U\mathbf{V}_B$$

$$(7.4b) \quad \mathbf{A}^H(\hat{\mathbf{U}}\mathbf{U}_B) = (\hat{\mathbf{V}}\mathbf{V}_B)\boldsymbol{\Sigma}_B^H + \mathbf{E}_V\mathbf{U}_B$$

This is an NPSVD where the coupling-matrix  $\boldsymbol{\Sigma}_B$  is diagonal by construction. Note that  $\|\tilde{\mathbf{E}}_U\| = \|\mathbf{E}_U\|$  and  $\|\tilde{\mathbf{E}}_V\| = \|\mathbf{E}_V\|$  in any unitarily invariant norm. As long as we can compute the SVD of  $\hat{\mathbf{B}}$ , there is no need to worry whether it is diagonal.

### NPSVD Diagonalization

Let  $\hat{\mathbf{B}} = \mathbf{U}_B \boldsymbol{\Sigma}_B \mathbf{V}_B^H$  be an SVD of  $\hat{\mathbf{B}}$ . The NPSVD can then be transformed to one with a diagonal coupling-matrix using the following equations:

$$(7.5) \quad \tilde{\mathbf{U}} = \hat{\mathbf{U}}\mathbf{U}_B \quad \tilde{\mathbf{B}} = \boldsymbol{\Sigma}_B \quad \tilde{\mathbf{V}} = \hat{\mathbf{V}}\mathbf{V}_B$$

$$(7.6) \quad \tilde{\mathbf{E}}_U = \mathbf{E}_U\mathbf{V}_B \quad \tilde{\mathbf{E}}_V = \mathbf{E}_V\mathbf{U}_B$$

When  $\hat{\mathbf{B}}$  is non-diagonal, the columns of  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  does not approximate singular vectors. Instead  $\text{range}(\hat{\mathbf{U}})$  and  $\text{range}(\hat{\mathbf{V}})$  approximate singular subspaces and the singular values of  $\hat{\mathbf{B}}$  approximate singular values of  $\mathbf{A}$ .

One should be warned that even though the singular values of  $\hat{\mathbf{B}}$  are accurate approximations to those of  $\mathbf{A}$  when the residual norms are small, the dimension of the invariant subspace can only be assessed when  $\hat{\mathbf{U}}^H \hat{\mathbf{U}} \approx \mathbf{I}$  and  $\hat{\mathbf{V}}^H \hat{\mathbf{V}} \approx \mathbf{I}$ . Otherwise several singular values of  $\hat{\mathbf{B}}$  might refer to the same singular value of  $\mathbf{A}$  (see sec. C7 on p. 153).

## 7.3 Updating the NPSVD

We must devise methods for reducing the residual norms in the NPSVD. There are two options at this point: either augment the singular matrices, to better approximate the residuals, or remove a part of the residuals. This is a rather crude explanation, since the operations are connected.

The latter is known as restarting and is discussed in sec. 7.7, while this section pursue the former strategy. Preliminary work showed that little is gained from

augmenting  $\text{range}(\hat{\mathbf{U}})$  and  $\text{range}(\hat{\mathbf{V}})$  separately, and we will only consider the double-update:

$$(7.7) \quad \tilde{\mathbf{U}} = [\hat{\mathbf{U}}, \hat{\mathbf{s}}_{\mathbf{u}}] \quad \tilde{\mathbf{V}} = [\hat{\mathbf{V}}, \hat{\mathbf{s}}_{\mathbf{v}}]$$

The vectors  $\hat{\mathbf{s}}_{\mathbf{u}}$  and  $\hat{\mathbf{s}}_{\mathbf{v}}$  represent directions that will become available for approximating the subspaces. With a terminology partly borrowed from [32], we shall refer to these vectors as *search vectors*.

### Updating the coupling matrix

We motivate the update by considering the case where  $\hat{\mathbf{U}}^H \hat{\mathbf{U}} = \mathbf{I}$  and  $\hat{\mathbf{V}}^H \hat{\mathbf{V}} = \mathbf{I}$ . In this case we may determine the matrix  $\hat{\mathbf{B}}$  that minimizes the residual norms. The least-squares approximation in theorem C8 states that  $\hat{\mathbf{B}} = \hat{\mathbf{U}}^+ \mathbf{A} \hat{\mathbf{V}}$  minimizes  $\|\mathbf{E}_{\mathbf{U}}\|$ , while  $\hat{\mathbf{B}}^H = \hat{\mathbf{V}}^+ \mathbf{A}^H \hat{\mathbf{U}}$  minimizes  $\|\mathbf{E}_{\mathbf{V}}\|$ . However, in this particular case the pseudo-inverses are simply the adjoint matrices, and the two equations for  $\hat{\mathbf{B}}$  are really the same:

$$(7.8) \quad \hat{\mathbf{B}} = \hat{\mathbf{U}}^H \mathbf{A} \hat{\mathbf{V}}$$

The updated NPSVD should also satisfy  $\tilde{\mathbf{B}} = \tilde{\mathbf{U}}^H \mathbf{A} \tilde{\mathbf{V}}$ . Inserting eq. (7.7) leads to:

$$(7.9) \quad \tilde{\mathbf{B}} = \begin{bmatrix} \hat{\mathbf{B}} & \hat{\mathbf{f}}_{\mathbf{v}} \\ \hat{\mathbf{f}}_{\mathbf{u}}^H & \beta \end{bmatrix}$$

where  $\hat{\mathbf{f}}_{\mathbf{u}}^H = \hat{\mathbf{s}}_{\mathbf{u}}^H \mathbf{A} \hat{\mathbf{V}}$ ,  $\hat{\mathbf{f}}_{\mathbf{v}} = \hat{\mathbf{U}}^H \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}}$ , and  $\beta = \hat{\mathbf{s}}_{\mathbf{u}}^H \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}}$ .

Even when the NPSVD is non-orthogonal, it remains helpful to keep the update of  $\tilde{\mathbf{B}}$  in this form. Therefore we use eq. (7.9) as it stands, but revise the updates of  $\hat{\mathbf{f}}_{\mathbf{u}}$ ,  $\hat{\mathbf{f}}_{\mathbf{v}}$ , and  $\beta$  in the general case.

### Updating the residual matrices

The residual matrices can be evaluated directly from their definition in eq. (7.3):

$$(7.10a) \quad \begin{aligned} \tilde{\mathbf{E}}_{\mathbf{U}} &= \mathbf{A} \tilde{\mathbf{V}} - \tilde{\mathbf{U}} \tilde{\mathbf{B}} \\ &= [\mathbf{A} \hat{\mathbf{V}}, \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}}] - [\hat{\mathbf{U}}, \hat{\mathbf{s}}_{\mathbf{v}}] \tilde{\mathbf{B}} \\ &= [(\mathbf{A} \hat{\mathbf{V}} - \hat{\mathbf{U}} \hat{\mathbf{B}}) - \hat{\mathbf{s}}_{\mathbf{u}} \hat{\mathbf{f}}_{\mathbf{u}}^H, \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}} - \hat{\mathbf{U}} \hat{\mathbf{f}}_{\mathbf{v}} - \hat{\mathbf{s}}_{\mathbf{u}} \beta] \\ &= (\mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}} - \hat{\mathbf{U}} \hat{\mathbf{f}}_{\mathbf{v}} - \hat{\mathbf{s}}_{\mathbf{u}} \beta) [0, \dots, 0, 1] + [\mathbf{E}_{\mathbf{U}} - \hat{\mathbf{s}}_{\mathbf{u}} \hat{\mathbf{f}}_{\mathbf{u}}^H, \mathbf{0}] \end{aligned}$$

Carrying out the same steps for  $\tilde{\mathbf{E}}_{\mathbf{V}}$  gives similarly:

$$(7.10b) \quad \tilde{\mathbf{E}}_{\mathbf{V}} = (\mathbf{A}^H \hat{\mathbf{s}}_{\mathbf{u}} - \hat{\mathbf{V}} \hat{\mathbf{f}}_{\mathbf{u}} - \hat{\mathbf{s}}_{\mathbf{v}} \beta^*) [0, \dots, 0, 1] + [\mathbf{E}_{\mathbf{V}} - \hat{\mathbf{s}}_{\mathbf{v}} \hat{\mathbf{f}}_{\mathbf{v}}^H, \mathbf{0}]$$

Thus, if we interpret  $\hat{\mathbf{s}}_{\mathbf{u}} \hat{\mathbf{f}}_{\mathbf{u}}^H \approx \mathbf{E}_{\mathbf{U}}$  as a rank one approximation, then the updated residual consists of the difference,  $\mathbf{E}_{\mathbf{U}} - \hat{\mathbf{s}}_{\mathbf{u}} \hat{\mathbf{f}}_{\mathbf{u}}^H$ , with an extra column appended. We may write this as an update of the search-vectors, which we scale so the updated search-vector is unit length.

### NPSVD Update

Given rank-one approximation of the residual matrices,  $\mathbf{E}_U \approx \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H$  and  $\mathbf{E}_V \approx \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H$ , and a constant,  $\beta$ , the following is an updated NPSVD:

$$(7.11) \quad \tilde{\mathbf{U}} = [\hat{\mathbf{U}}, \hat{\mathbf{s}}_u] \quad \tilde{\mathbf{V}} = [\hat{\mathbf{V}}, \hat{\mathbf{s}}_v]$$

$$(7.12) \quad \tilde{\mathbf{B}} = \begin{bmatrix} \mathbf{B} & \hat{\mathbf{f}}_v \\ \hat{\mathbf{f}}_u^H & \beta \end{bmatrix}$$

$$(7.13) \quad \tilde{\mathbf{s}}_u = (\mathbf{A} \hat{\mathbf{s}}_v - \hat{\mathbf{U}} \hat{\mathbf{f}}_v - \hat{\mathbf{s}}_u \beta) / \tilde{f}_u \quad \tilde{\mathbf{s}}_v = (\mathbf{A}^H \hat{\mathbf{s}}_u - \hat{\mathbf{V}} \hat{\mathbf{f}}_u - \hat{\mathbf{s}}_v \beta^*) / \tilde{f}_v$$

$$(7.14) \quad \tilde{\mathbf{f}}_u = \begin{bmatrix} \mathbf{0} \\ \tilde{f}_u \end{bmatrix} \quad \tilde{\mathbf{f}}_v = \begin{bmatrix} \mathbf{0} \\ \tilde{f}_v \end{bmatrix}$$

$$(7.15) \quad \tilde{\mathbf{E}}_U = \tilde{\mathbf{s}}_u \tilde{\mathbf{f}}_u^H + [\mathbf{E}_U - \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H, \mathbf{0}] \quad \tilde{\mathbf{E}}_V = \tilde{\mathbf{s}}_v \tilde{\mathbf{f}}_v^H + [\mathbf{E}_V - \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H, \mathbf{0}]$$

### Choosing search-vectors

The update in eq. (7.15) implies that  $\|\tilde{\mathbf{E}}_U\|_F^2 = \|\mathbf{E}_U - \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H\|_F^2 + \tilde{f}_u^2 \|\tilde{\mathbf{s}}_u\|_F^2$ . The goal is to minimize this quantity.

One way to proceed is the Jacobi-Davidson SVD [32]. This approach use a search-vector that keeps the norm small using an iterative procedure based on a preconditioner for  $\mathbf{A}$ . However, it may not always be possible to find a suitable preconditioner, which makes the scheme unsuitable for a canned routine.

A more pragmatic approach is to minimize the first term by setting  $\hat{\mathbf{s}}_u$  as the first left singular vector of  $\mathbf{E}_U$  (from the low-rank approximation in theorem C9). Likewise,  $\hat{\mathbf{s}}_v$  should be the first left singular vector of  $\mathbf{E}_V$ . In practice the search-vectors are generated iteratively.

### Rounding errors

It is worthwhile to consider the effect of rounding errors in some detail. If the computed matrix-vector product satisfies  $\|f(\mathbf{A}, \hat{\mathbf{s}}_v) - \mathbf{A} \hat{\mathbf{s}}_v\|_2 \leq c_1 \|\mathbf{A}\|_2 \cdot \|\hat{\mathbf{s}}_v\|_2$  evaluating eq. (7.15) in floating-point leads to:

$$\|\tilde{\mathbf{E}}_U - \tilde{\mathbf{s}}_u \tilde{\mathbf{f}}_u^H\|_F \leq \|\mathbf{E}_U - \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H\|_F + c_1 \|\mathbf{A}\|_2 + (c_2 \|\hat{\mathbf{U}}\|_2) \|\hat{\mathbf{f}}_v\|_2 + \epsilon |\beta|$$

While  $c_1$  may differ between applications,  $c_2$  is usually near machine-epsilon. Now consider running  $k$  update steps after an initialization. The term  $\|\hat{\mathbf{U}}\|_2$  can be dropped since only the last element of  $\hat{\mathbf{f}}_v$  is non-zero. Noting that  $\hat{\mathbf{f}}_v$  and  $\beta$  are embedded in the updated coupling matrix leads to:

$$\|\tilde{\mathbf{E}}_U - \tilde{\mathbf{s}}_u \tilde{\mathbf{f}}_u^H\|_F \leq k c_1 \|\mathbf{A}\|_2 + c_2 \|\tilde{\mathbf{B}}\|_F$$

In practical algorithms the singular values of  $\tilde{\mathbf{B}}$  approximates those of  $\mathbf{A}$ , and we may consider  $\|\tilde{\mathbf{E}}_U - \tilde{\mathbf{s}}_u \tilde{\mathbf{f}}_u^H\|_F \leq k(c_1 + c_2) \|\mathbf{A}\|_2$  to be an upper bound. This implies that the quality of estimated singular triplets is limited by the matrix-vector product, not accumulation of rounding errors.

### Algorithms



To illustrate the development, several example implementations will be sprinkled throughout this chapter. It must be emphasized that they are meant for demonstration only (see sec. 1.7 p. 6).

Algorithm 7.1 creates an NPSVD from two starting-vectors, and uses eq. (7.8) for the coupling-matrix.

The update in alg. 7.3 is more complicated. Inspecting eq. (7.13) we see that only the last component is non-zero. Since the majority of work in the update is matrix multiplications with  $\mathbf{A}$ ,  $\hat{\mathbf{U}}$ , and  $\hat{\mathbf{V}}$  it is important to reduce this work. Therefore the algorithm allows  $\hat{f}_u$  and  $\hat{f}_v$  to be scalars, and also returns scalar values.

## 7.4 Updates with reorthogonalization

Most software packages keep the columns of the singular matrices  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  (nearly) orthogonal during the update. This material would be incomplete without

### Algorithm 7.1: Initialize NPSVD

Create an NPSVD from two start-vectors.

---

```

function [ $\tilde{\mathbf{U}}, \tilde{\mathbf{V}}, \tilde{\mathbf{B}}, \tilde{s}_u, \tilde{s}_v, \tilde{f}_u, \tilde{f}_v$ ] = init( $\mathbf{A}, \hat{\mathbf{u}}, \hat{\mathbf{v}}$ )
     $\tilde{\mathbf{U}} = \hat{\mathbf{u}} / \|\hat{\mathbf{u}}\|_2$ ;    $\tilde{\mathbf{V}} = \hat{\mathbf{v}} / \|\hat{\mathbf{v}}\|_2$ ;           Normalize starting vectors
     $\mathbf{x}_u = \mathbf{A} * \tilde{\mathbf{V}}$ ;    $\mathbf{x}_v = \mathbf{A}^H * \tilde{\mathbf{U}}$ ;           Form products
     $\tilde{\mathbf{B}} = \tilde{\mathbf{U}}^H * \mathbf{x}_u$ ;           Coupling-matrix by eq. (7.8)
    [ $\tilde{s}_u, \tilde{f}_u$ ] = unit( $\mathbf{x}_u - \tilde{\mathbf{U}} * \tilde{\mathbf{B}}$ );           Left residual by eq. (7.3a)
    [ $\tilde{s}_v, \tilde{f}_v$ ] = unit( $\mathbf{x}_v - \tilde{\mathbf{V}} * \tilde{\mathbf{B}}^H$ );           Right residual by eq. (7.3b)

```

---

### Algorithm 7.2: Normalize a vector

Auxiliary routine that factorizes a vector,  $\mathbf{x} = \alpha\mathbf{y}$ , where  $\mathbf{y}$  has unit length. The case  $\mathbf{x} = \mathbf{0}$  occurs surprisingly often in applications, even in floating-point arithmetics.

---

```

function [ $\mathbf{y}, \alpha$ ] = unit( $\mathbf{x}$ )
     $\alpha = \|\mathbf{x}\|_2$ ;           Compute norm
    if  $\alpha \equiv 0$ 
         $\mathbf{y} = \text{randn}(\text{size}(\mathbf{x}))$ ;    $\mathbf{y} = \mathbf{y} / \|\mathbf{y}\|_2$ ;           Create random unit vector
    else
         $\mathbf{y} = \mathbf{x} / \alpha$ ;           Normalize input vector

```

---

### Algorithm 7.3: Update NPSVD

Update a NPSVD as described on p. 84. If necessary, the search-vectors are prepended with zeros to avoid excessive work.

---

```

function [ $\tilde{\mathbf{U}}, \tilde{\mathbf{V}}, \tilde{\mathbf{B}}, \tilde{s}_u, \tilde{s}_v, \tilde{f}_u, \tilde{f}_v$ ] = update( $\mathbf{A}, \hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{s}_u, \hat{s}_v, \hat{f}_u, \hat{f}_v, \beta$ )
     $\tilde{\mathbf{V}} = [\hat{\mathbf{V}}, \hat{s}_v]$ ;    $\tilde{\mathbf{U}} = [\hat{\mathbf{U}}, \hat{s}_u]$ ;           Update singular matrices by eq. (7.7)
     $l_u = \text{size}(\hat{\mathbf{B}}, 1) - \text{length}(\hat{f}_u)$ ;    $\mathbf{z}_u = \text{zeros}(l_u, 1)$ ;           Form implicit zeros in  $\mathbf{f}_u$ 
     $l_v = \text{size}(\hat{\mathbf{B}}, 2) - \text{length}(\hat{f}_v)$ ;    $\mathbf{z}_v = \text{zeros}(l_v, 1)$ ;           And for  $\mathbf{f}_v$ 
     $\tilde{\mathbf{B}} = [\hat{\mathbf{B}}, [\mathbf{z}_v; \hat{f}_v]; [\mathbf{z}_u; \hat{f}_u]^H, \beta]$ ;           Update coupling-matrix using eq. (7.12)
     $\tilde{s}_u = \mathbf{A} * \hat{s}_v - \tilde{\mathbf{U}}(:, l_v + 1 : \text{end}) * \hat{f}_v - \hat{s}_u * \beta$ ;           Unscaled search-vector from eq. (7.10a)
     $\tilde{s}_v = \mathbf{A}^H * \hat{s}_u - \tilde{\mathbf{V}}(:, l_u + 1 : \text{end}) * \hat{f}_u - \hat{s}_v * (\beta)^*$ ;           Unscaled search-vector from eq. (7.10b)
    [ $\tilde{s}_u, \tilde{f}_u$ ] = unit( $\tilde{s}_u$ );           Normalize search-vector
    [ $\tilde{s}_v, \tilde{f}_v$ ] = unit( $\tilde{s}_v$ );           Normalize search-vector

```

---

discussing this, even if the mechanism in sec. 7.6 is employed.

Assume that  $\hat{\mathbf{U}}^H \hat{\mathbf{U}} = \mathbf{I}$  and  $\hat{\mathbf{V}}^H \hat{\mathbf{V}} = \mathbf{I}$ . It was shown in sec. 7.3.1 the update that minimizes any unitarily invariant norm of the residuals is  $\hat{\mathbf{f}}_{\mathbf{u}} = \hat{\mathbf{V}}^H \mathbf{A}^H \hat{\mathbf{s}}_{\mathbf{u}}$ ,  $\hat{\mathbf{f}}_{\mathbf{v}} = \hat{\mathbf{U}}^H \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}}$ , and  $\beta = \hat{\mathbf{s}}_{\mathbf{u}}^H \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}}$ . Inserting this into the formula for the updated search-vector, eq. (7.13), gives:

$$\begin{aligned} \tilde{\mathbf{s}}_{\mathbf{u}} \tilde{f}_{\mathbf{u}} &= \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}} - \hat{\mathbf{U}} \hat{\mathbf{f}}_{\mathbf{v}} - \hat{\mathbf{s}}_{\mathbf{u}} \beta \\ &= \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}} - \hat{\mathbf{U}} \hat{\mathbf{U}}^H \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}} - \hat{\mathbf{s}}_{\mathbf{u}} \hat{\mathbf{s}}_{\mathbf{u}}^H \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}} \\ &= (\mathbf{I} - \tilde{\mathbf{U}} \tilde{\mathbf{U}}^H) \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}} \end{aligned} \tag{7.16a}$$

$$\tilde{\mathbf{s}}_{\mathbf{v}} \tilde{f}_{\mathbf{v}} = (\mathbf{I} - \tilde{\mathbf{V}} \tilde{\mathbf{V}}^H) \mathbf{A}^H \hat{\mathbf{s}}_{\mathbf{u}} \tag{7.16b}$$

If we assume that the search-vectors are orthogonal to the singular matrices,  $\hat{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}} = \hat{\mathbf{V}}^H \hat{\mathbf{s}}_{\mathbf{v}} = \mathbf{0}$ , then it follows that the updated search-vectors are orthogonal to the updated singular matrices, i.e.  $\tilde{\mathbf{U}}^H \tilde{\mathbf{s}}_{\mathbf{u}} = \tilde{\mathbf{V}}^H \tilde{\mathbf{s}}_{\mathbf{v}} = \mathbf{0}$ . Unfortunately, this holds only in exact arithmetic; a major cause of algorithmic inefficiency.

Instead an algorithm can enforce the property by orthogonalizing the updated search-vectors against the singular matrices. Informally, orthogonalization removes a component of the search-vector, say  $\hat{\mathbf{s}}_{\mathbf{u}}$ , in the range of  $\hat{\mathbf{U}}$  which is a result of rounding errors.

An algorithm that orthogonalizes the search-vectors every time they are updated is said perform “full reorthogonalization”. This is sometimes used to approximate the behavior of an algorithm that runs in exact arithmetic [15].

It becomes expensive to use full reorthogonalization if the iteration is allowed to run for many steps. Instead it is possible to reorthogonalize either occasionally with all vectors, or with some previous search-vectors at each step.

This strategy is known as *selective reorthogonalization*. Usually the vectors are kept *semi-orthogonal* so that  $|\hat{\mathbf{s}}_{\mathbf{u}}^H \hat{\mathbf{u}}_i| \leq \sqrt{\epsilon_M}$  for  $i = 1, \dots, k$  [15].

## 7.5 Krylov algorithms

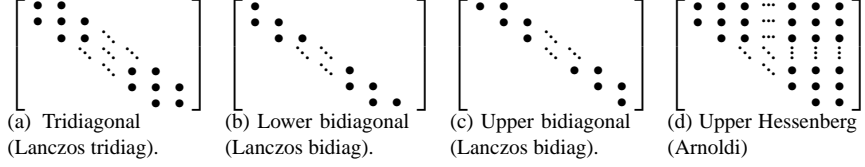
The Lanczos- and the Arnoldi-iterations are classical algorithms for large-scale SVD- and eigenvalue-problems. Each algorithm result from a specific choice of the search-vectors  $\hat{\mathbf{s}}_{\mathbf{u}}$  and  $\hat{\mathbf{s}}_{\mathbf{v}}$ .

We will develop the algorithms by keeping the residual matrices,  $\mathbf{E}_{\mathbf{U}}$  and  $\mathbf{E}_{\mathbf{V}}$ , at rank one during (almost) all steps. As discussed in sec. 7.3.4 an update (i.e. alg. 7.3) returns a useful approximations of the residual matrices, even in the presence of rounding errors. For completeness we mention that there exist block-algorithms using residuals with higher ranks [26].

Usually the classical algorithms are developed in terms of Krylov subspaces, e.g. [27, 73]. In this case the Krylov subspace  $\text{range}(\mathbf{K}_n)$ , where the *Krylov matrix* is given by  $\mathbf{K}_n = [\mathbf{x}, \mathbf{A}\mathbf{x}, \mathbf{A}^2\mathbf{x}, \dots, \mathbf{A}^{n-1}\mathbf{x}]$ , is taken as an “obvious place” to search.

The idea of minimizing the residual norms is rarely made explicit, although some derivations resembles those in the previous sections superficially, e.g. [67, p.8]. Instead the algorithms are derived by reducing the Krylov-matrix to a prescribed form of the coupling-matrix  $\hat{\mathbf{B}}$ , as shown in fig. 7.2.

**Figure 7.2:**  
Structure of  $\hat{\mathbf{B}}$   
generated by the  
Krylov-algorithms.  
Only “•”-elements  
can be nonzero.



## Lanczos tridiagonalization

The Lanczos iteration attempts to find a partial SVD. It starts with two vectors and attempts to reduce the residual norms by repeatedly updating the NPSVD with the updated search-vectors.

In principle  $\beta$  can be chosen freely, and the first idea is probably to set  $\beta = 0$ . However, since the goal is to minimize the residual norms it is better to follow sec. 7.3.1 and use  $\beta = \hat{\mathbf{s}}_{\mathbf{u}}^{\mathbf{H}} \mathbf{A} \hat{\mathbf{s}}_{\mathbf{v}}$ . This leads directly to algorithm 7.4.

Note that  $\hat{\mathbf{f}}_{\mathbf{u}}$  and  $\hat{\mathbf{f}}_{\mathbf{v}}$  only contains a single non-zero entry each, so  $\hat{\mathbf{B}}$  is tridiagonal. However, rather than being the starting-point, this structure of the coupling-matrix is a byproduct of the desire to compute singular subspaces.

## Lanczos bidiagonalization

One can do better than keeping both errors at rank one. If we are given  $\hat{\mathbf{U}}$  then we can choose  $\hat{\mathbf{V}}$  such that  $\text{range}(\hat{\mathbf{V}}) = \text{range}(\mathbf{A}^{\mathbf{H}} \hat{\mathbf{U}})$ , e.g. using a QR-factorization. This allows us to set  $\mathbf{E}_{\mathbf{V}} \approx \mathbf{0}$ .

If  $\mathbf{E}_{\mathbf{V}} \approx \hat{\mathbf{s}}_{\mathbf{v}} \hat{\mathbf{f}}_{\mathbf{v}}^{\mathbf{H}} = \mathbf{0}$ , then we may choose  $\hat{\mathbf{s}}_{\mathbf{v}}$  freely. Since we want a recurrence where  $\tilde{\mathbf{E}}_{\mathbf{V}} \approx \mathbf{0}$ , it follows from eq. (7.13) that we should choose  $\hat{\mathbf{s}}_{\mathbf{v}} \beta^* = \mathbf{A}^{\mathbf{H}} \hat{\mathbf{s}}_{\mathbf{u}} - \hat{\mathbf{V}} \hat{\mathbf{f}}_{\mathbf{u}}$ . This immediately leads to algorithm 7.5.

The bidiagonal structure of the coupling-matrix follows from the fact that  $\hat{\mathbf{f}}_{\mathbf{v}} = \mathbf{0}$  and  $\hat{\mathbf{f}}_{\mathbf{u}}$  contains a single non-zero element. Keeping  $\mathbf{E}_{\mathbf{U}} \approx \hat{\mathbf{s}}_{\mathbf{u}} \hat{\mathbf{f}}_{\mathbf{u}}^{\mathbf{H}} = \mathbf{0}$  instead would similarly lead to an upper bidiagonal coupling-matrix.

### Algorithm 7.4: Lanczos tridiagonalization

*The Lanczos tridiagonalization algorithm in terms of NPSVD updates.*

```

function [ $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{f}_{\mathbf{u}}, \hat{f}_{\mathbf{v}}$ ] = tridiag( $\mathbf{A}, \hat{\mathbf{u}}, \hat{\mathbf{v}}, N$ )
    [ $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{f}_{\mathbf{u}}, \hat{f}_{\mathbf{v}}$ ] = init( $\mathbf{A}, \hat{\mathbf{u}}, \hat{\mathbf{v}}$ );
    for  $n = 1 : N - 1$ 
         $\beta = \hat{\mathbf{s}}_{\mathbf{u}}^{\mathbf{H}} * \mathbf{A} * \hat{\mathbf{s}}_{\mathbf{v}}$ ;
        [ $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{f}_{\mathbf{u}}, \hat{f}_{\mathbf{v}}$ ] = update( $\mathbf{A}, \hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{f}_{\mathbf{u}}, \hat{f}_{\mathbf{v}}, \beta$ );

```

Expand to  $N$  columns  
Good choice, but could be anything

### Algorithm 7.5: Lanczos bidiagonalization

*The Lanczos lower bidiagonalization algorithm in terms of NPSVD updates.*

```

function [ $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{f}_{\mathbf{u}}, \hat{f}_{\mathbf{v}}$ ] = bidiag( $\mathbf{A}, \hat{\mathbf{u}}, N$ )
    [ $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{f}_{\mathbf{u}}, \hat{f}_{\mathbf{v}}$ ] = init( $\mathbf{A}, \hat{\mathbf{u}}, \mathbf{A}^{\mathbf{H}} * \hat{\mathbf{u}}$ );
    for  $n = 1 : N - 1$ 
         $\hat{\mathbf{s}}_{\mathbf{v}} = \mathbf{A}^{\mathbf{H}} * \hat{\mathbf{s}}_{\mathbf{u}} - \hat{\mathbf{V}}(:, n) * \hat{f}_{\mathbf{u}}$ ;
        [ $\hat{\mathbf{s}}_{\mathbf{v}}, \alpha$ ] = unit( $\hat{\mathbf{s}}_{\mathbf{v}}$ );  $\beta = (\alpha)^*$ ;
        [ $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{f}_{\mathbf{u}}, \hat{f}_{\mathbf{v}}$ ] = update( $\mathbf{A}, \hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{f}_{\mathbf{u}}, 0, \beta$ );

```

Expand to  $N$  columns  
Search-vector that keep  $\tilde{\mathbf{E}}_{\mathbf{V}} = \mathbf{0}$   
Scale

## The Arnoldi iteration

The Arnoldi iteration is used in eigenvalue-problems, which is a digression from the topic of this chapter. We include it for completeness and because it is important in practice.

The goal is to find a unitary matrix,  $\hat{\mathbf{Q}}$ , that (nearly) spans an invariant subspace, i.e.  $\mathbf{A}\hat{\mathbf{Q}} = \hat{\mathbf{Q}}\hat{\mathbf{B}} + \mathbf{E}$ ,  $\mathbf{E} \approx \mathbf{0}$ . Inserting a Schur-decomposition of  $\hat{\mathbf{B}}$  leads to an approximate Schur-decomposition of  $\mathbf{A}$ . The hope is to find an acceptable approximation while the dimension of  $\hat{\mathbf{B}}$  is small enough to allow this computation.

Comparing with the NPSVD in eq. (7.3) we see that the desired relation is obtained for  $\hat{\mathbf{Q}} = \hat{\mathbf{U}} = \hat{\mathbf{V}}$  and  $\mathbf{E}_{\hat{\mathbf{U}}} = \mathbf{E}$ . To keep  $\tilde{\mathbf{U}} = \tilde{\mathbf{V}}$  it is necessary to set  $\hat{\mathbf{s}}_{\mathbf{v}} = \hat{\mathbf{s}}_{\mathbf{u}}$ . We obtain a recurrence that produces orthogonal search-vectors by setting  $\hat{\mathbf{f}}_{\mathbf{v}}$  and  $\beta$  so  $\tilde{\mathbf{s}}_{\mathbf{u}}\hat{\mathbf{f}}_{\mathbf{u}} = \mathbf{A}\hat{\mathbf{s}}_{\mathbf{v}} - \tilde{\mathbf{U}}\hat{\mathbf{f}}_{\mathbf{v}} - \hat{\mathbf{s}}_{\mathbf{u}}\beta$  is orthogonal to  $\tilde{\mathbf{U}}$ .

Algorithm 7.6 implements the idea using Gram-Schmidt orthogonalization. Since  $\hat{\mathbf{f}}_{\mathbf{u}}$  only contains a single non-zero entry, it follows that  $\hat{\mathbf{B}}$  is upper Hessenberg.

This formulation reveals that  $\|\mathbf{E}_{\hat{\mathbf{V}}}\|$  is uncontrolled. Since  $\|\mathbf{E}_{\hat{\mathbf{V}}} - \hat{\mathbf{s}}_{\mathbf{v}}\hat{\mathbf{f}}_{\mathbf{v}}^{\mathbf{H}}\|$  can be arbitrarily large, so can  $\|\tilde{\mathbf{E}}_{\hat{\mathbf{V}}}\|$ . As the precision of the eigenvalue problem depends on both left and right eigenvectors [27], this might be a problem. Since the Arnoldi iteration is of no use in SVD computations, this issue will not be pursued further.

001010  
100001  
111100

## Algorithms

The implementations in alg. 7.4–7.6 are unusual, and at first sight alg. 7.5 and its traditional counterpart, e.g. bidiag 1 in [55], might appear to be different. It is nevertheless trivial to optimize the NPSVD-based algorithms to their traditional version.

Lanczos tridiagonalization, as given by alg. 7.4, is very general. Normally it is applied to Hermitian matrices with a single starting-vector used twice. In this case alg. 7.4 can be considered as a special variant of the Arnoldi iteration.

One should not mistake alg. 7.4 with “unsymmetric Lanczos tridiagonalization” [26]. Although the coupling matrix is updated in a similar way, the search-vectors

### Algorithm 7.6: Arnoldi reduction to upper Hessenberg form.

*The Arnoldi iteration written in terms of a NPSVD.*

---

```

function [ $\hat{\mathbf{Q}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{f}}_{\mathbf{u}}$ ] = arnoldi( $\mathbf{A}, \mathbf{x}, N$ )
  [ $\hat{\mathbf{Q}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{\mathbf{f}}_{\mathbf{u}}, \hat{\mathbf{f}}_{\mathbf{v}}$ ] = init( $\mathbf{A}, \mathbf{x}, \mathbf{x}$ );
  for  $n = 1 : N - 1$ 
     $\hat{\mathbf{s}}_{\mathbf{v}} = \hat{\mathbf{s}}_{\mathbf{u}}; \tilde{\mathbf{s}}_{\mathbf{u}} = \mathbf{A} * \hat{\mathbf{s}}_{\mathbf{v}};$ 
    for  $j = 1 : n$ 
       $\hat{\mathbf{f}}_{\mathbf{v}}(j, 1) = \hat{\mathbf{Q}}(:, j)^{\mathbf{H}} * \tilde{\mathbf{s}}_{\mathbf{u}};$ 
       $\tilde{\mathbf{s}}_{\mathbf{u}} = \tilde{\mathbf{s}}_{\mathbf{u}} - \hat{\mathbf{Q}}(:, j) * \hat{\mathbf{f}}_{\mathbf{v}}(j, 1);$ 
    end
     $\beta = \hat{\mathbf{s}}_{\mathbf{u}}^{\mathbf{H}} * \tilde{\mathbf{s}}_{\mathbf{u}};$ 
     $\tilde{\mathbf{s}}_{\mathbf{u}} = \tilde{\mathbf{s}}_{\mathbf{u}} - \hat{\mathbf{s}}_{\mathbf{u}} * \beta;$ 
    [ $\hat{\mathbf{Q}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{v}}, \hat{\mathbf{f}}_{\mathbf{u}}, \hat{\mathbf{f}}_{\mathbf{v}}$ ] = update( $\mathbf{A}, \hat{\mathbf{Q}}, \hat{\mathbf{Q}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{s}}_{\mathbf{u}}, \hat{\mathbf{f}}_{\mathbf{u}}, \hat{\mathbf{f}}_{\mathbf{v}}, \beta$ );
  end

```

---

Iterate  
Ensure  $\mathbf{s}_{\mathbf{u}} = \mathbf{s}_{\mathbf{v}}$  and form the product  
Make  $\tilde{\mathbf{s}}_{\mathbf{u}}$  orthogonal to  $\tilde{\mathbf{U}}$   
  
Make  $\tilde{\mathbf{s}}_{\mathbf{u}}$  orthogonal to  $\tilde{\mathbf{U}}$   
Needlessly recomputed in *update*

are quite different. While unsymmetric Lanczos is an unstable method for eigenvalue problems, alg. 7.4 is perfectly stable for SVD-problems.

Starting from MATLAB<sup>®</sup> 6.0, the function `eigs` is based on the Arnoldi iteration implemented in ARPACK [50]. In turn, `svds` for computing a PSVD, is based on `eigs`. However, the matrix involved is Hermitian and ARPACK uses Lanczos tridiagonalization, not Arnoldi.

## 7.6 Reorthogonalization

Just as we could restructure  $\hat{\mathbf{B}}$  by diagonalizing it, it is also possible to reorthogonalize an NPSVD. Instead of keeping the columns of  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  nearly orthogonal, it will suffice to keep them well-conditioned.

Let  $\hat{\mathbf{U}} = \tilde{\mathbf{U}}\mathbf{R}_U$  and  $\hat{\mathbf{V}} = \tilde{\mathbf{V}}\mathbf{R}_V$  be QR-decompositions. The coupling-matrix that minimizes the residual norms is given in sec. 7.3.1 on p. 83:

$$\begin{aligned}\tilde{\mathbf{B}} &= \tilde{\mathbf{U}}^H \mathbf{A} \tilde{\mathbf{V}} \\ &= (\hat{\mathbf{U}}\mathbf{R}_U^{-1})^H \mathbf{A} \hat{\mathbf{V}}\mathbf{R}_V^{-1} \\ &= \mathbf{R}_U^{-H} (\hat{\mathbf{U}}^H \mathbf{A} \hat{\mathbf{V}}) \mathbf{R}_V^{-1}\end{aligned}$$

Inserting  $\mathbf{A}\hat{\mathbf{V}} = \hat{\mathbf{U}}\hat{\mathbf{B}} + \mathbf{E}_U$  from eq. (7.3a) yields

$$\begin{aligned}\tilde{\mathbf{B}} &= \mathbf{R}_U^{-H} (\hat{\mathbf{U}}^H (\hat{\mathbf{U}}\hat{\mathbf{B}} + \mathbf{E}_U)) \mathbf{R}_V^{-1} \\ &= \mathbf{R}_U^{-H} (\mathbf{R}_U^H \mathbf{R}_U \hat{\mathbf{B}} + \hat{\mathbf{U}}^H \mathbf{E}_U) \mathbf{R}_V^{-1} \\ &= \mathbf{R}_U \hat{\mathbf{B}} \mathbf{R}_V^{-1} + \tilde{\mathbf{U}}^H \mathbf{E}_U \mathbf{R}_V^{-1}\end{aligned}\tag{7.17a}$$

or, if we insert  $\hat{\mathbf{U}}^H \mathbf{A} = \hat{\mathbf{B}}\hat{\mathbf{V}}^H + \mathbf{E}_V^H$  from eq. (7.3b) instead,

$$\tilde{\mathbf{B}} = \mathbf{R}_U^{-H} \hat{\mathbf{B}} \mathbf{R}_V^H + \mathbf{R}_U^{-H} \mathbf{E}_V^H \tilde{\mathbf{V}}\tag{7.17b}$$

The new residuals follows from their definition in eq. (7.3) by inserting eq. (7.17b) in eq. (7.17a) and eq. (7.18b):

$$\tilde{\mathbf{E}}_U = \mathbf{A}\tilde{\mathbf{V}} - \tilde{\mathbf{U}}\tilde{\mathbf{B}} = (\mathbf{I} - \tilde{\mathbf{U}}\tilde{\mathbf{U}}^H) \mathbf{E}_U \mathbf{R}_V^{-1}\tag{7.18a}$$

$$\tilde{\mathbf{E}}_V = \mathbf{A}^H \tilde{\mathbf{U}} - \tilde{\mathbf{V}}\tilde{\mathbf{B}}^H = (\mathbf{I} - \tilde{\mathbf{V}}\tilde{\mathbf{V}}^H) \mathbf{E}_V \mathbf{R}_U^{-1}\tag{7.18b}$$

Assume we have residual approximations  $\mathbf{E}_U \approx \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H$  and  $\mathbf{E}_V \approx \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H$ . Adding and subtracting this to the residuals gives:

$$\begin{aligned}\tilde{\mathbf{E}}_U &= (\mathbf{I} - \tilde{\mathbf{U}}\tilde{\mathbf{U}}^H) (\mathbf{E}_U - \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H) \mathbf{R}_V^{-1} + (\mathbf{I} - \tilde{\mathbf{U}}\tilde{\mathbf{U}}^H) \hat{\mathbf{s}}_u (\hat{\mathbf{f}}_u^H \mathbf{R}_V^{-1}) \\ \tilde{\mathbf{E}}_V &= (\mathbf{I} - \tilde{\mathbf{V}}\tilde{\mathbf{V}}^H) (\mathbf{E}_V - \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H) \mathbf{R}_U^{-1} + (\mathbf{I} - \tilde{\mathbf{V}}\tilde{\mathbf{V}}^H) \hat{\mathbf{s}}_v (\hat{\mathbf{f}}_v^H \mathbf{R}_U^{-1})\end{aligned}$$

The last term is the transform equations for the search-vectors:

$$\begin{aligned}\tilde{\mathbf{s}}_u &= (\mathbf{I} - \tilde{\mathbf{U}}\tilde{\mathbf{U}}^H) \hat{\mathbf{s}}_u & \tilde{\mathbf{f}}_u^H &= \hat{\mathbf{f}}_u^H \mathbf{R}_V^{-1} \\ \tilde{\mathbf{s}}_v &= (\mathbf{I} - \tilde{\mathbf{V}}\tilde{\mathbf{V}}^H) \hat{\mathbf{s}}_v & \tilde{\mathbf{f}}_v^H &= \hat{\mathbf{f}}_v^H \mathbf{R}_U^{-1}\end{aligned}$$

Since  $\tilde{\mathbf{U}}$  and  $\tilde{\mathbf{V}}$  have orthonormal columns, we have the satisfying result that the updated search-vectors are orthogonal to the updated singular matrices.

**NPSVD Reorthogonalization**

Given rank-1 approximations,  $\mathbf{E}_U \approx \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H$  and  $\mathbf{E}_V \approx \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H$ , the following returns an orthogonalized NPSVD.

Let  $\hat{\mathbf{U}} = \tilde{\mathbf{U}}\mathbf{R}_U$  and  $\hat{\mathbf{V}} = \tilde{\mathbf{V}}\mathbf{R}_V$  be QR-decompositions:

$$(7.19) \quad \tilde{\mathbf{s}}_u = (\mathbf{I} - \tilde{\mathbf{U}}\tilde{\mathbf{U}}^H)\hat{\mathbf{s}}_u \quad \tilde{\mathbf{s}}_v = (\mathbf{I} - \tilde{\mathbf{V}}\tilde{\mathbf{V}}^H)\hat{\mathbf{s}}_v$$

$$(7.20) \quad \tilde{\mathbf{f}}_u = \mathbf{R}_U^{-H}\hat{\mathbf{f}}_u \quad \tilde{\mathbf{f}}_v = \mathbf{R}_V^{-H}\hat{\mathbf{f}}_v$$

$$(7.21) \quad \tilde{\mathbf{B}} = \begin{cases} \mathbf{R}_U \hat{\mathbf{B}} \mathbf{R}_V^{-1} + (\tilde{\mathbf{U}}^H \hat{\mathbf{s}}_u) \tilde{\mathbf{f}}_u^H & \text{or} \\ \mathbf{R}_U^{-H} \hat{\mathbf{B}} \mathbf{R}_V^H + \tilde{\mathbf{f}}_v (\tilde{\mathbf{V}}^H \hat{\mathbf{s}}_v)^H \end{cases}$$

$$(7.22a) \quad \tilde{\mathbf{E}}_U = \tilde{\mathbf{s}}_u \tilde{\mathbf{f}}_u^H + (\mathbf{I} - \tilde{\mathbf{U}}\tilde{\mathbf{U}}^H)(\mathbf{E}_U - \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H) \mathbf{R}_U^{-1}$$

$$(7.22b) \quad \tilde{\mathbf{E}}_V = \tilde{\mathbf{s}}_v \tilde{\mathbf{f}}_v^H + (\mathbf{I} - \tilde{\mathbf{V}}\tilde{\mathbf{V}}^H)(\mathbf{E}_V - \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H) \mathbf{R}_V^{-1}$$

$$(7.22c)$$

**Rounding errors**

To analyze the growth of the residual norm, we first note that  $\|(\mathbf{I} - \tilde{\mathbf{U}}\tilde{\mathbf{U}}^H)\mathbf{X}\| \leq \|\mathbf{X}\|$  in any unitarily invariant norm (see theorem C7 on p. 148). It follows from eq. (7.18) that the residual norms can grow by at most a factor of  $\|\mathbf{R}_U^{-1}\|$  and  $\|\mathbf{R}_V^{-1}\|$ .

More specifically, consider the residual due to previous rounding errors:

$$(7.23a) \quad \|\tilde{\mathbf{E}}_U - \tilde{\mathbf{s}}_u \tilde{\mathbf{f}}_u^H\| \leq \|\mathbf{E}_U - \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H\| \cdot \|\mathbf{R}_U^{-1}\|$$

$$(7.23b) \quad \|\tilde{\mathbf{E}}_V - \tilde{\mathbf{s}}_v \tilde{\mathbf{f}}_v^H\| \leq \|\mathbf{E}_V - \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H\| \cdot \|\mathbf{R}_V^{-1}\|$$

Therefore the NPSVD remains accurate after the transformation, as long as  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  are well-conditioned. This is not nearly as strict as the semi-orthogonality condition described in sec. 7.4.

001010  
100001  
111100

**Algorithm**

A straightforward implementation is given in alg. 7.7. As the update described in sec. 7.3, it also simplifies in the case where  $\hat{\mathbf{f}}_u$  and  $\hat{\mathbf{f}}_v$  have a single non-zero entry. Since  $\mathbf{R}_U^{-H}$  is lower triangular it follows that  $\mathbf{R}_U^{-H}\hat{\mathbf{f}}_u$  also consists a single non-zero entry. In fact we have  $\tilde{f}_u = \hat{f}_u / (\mathbf{R}_U)_{(k,k)}^*$ .

**7.7 Restarting Lanczos**

When dealing with large-scale problems the strategies developed so far are insufficient. The main problem is that there is no way to know in advance how many steps are required before the residual norms are sufficiently small.

In fact we are supposed to deal with problems so large that not even the requested singular triplets can be held in memory at a single time. We also want to avoid disk storage, if at all possible.

The solution is to restart the procedure. When we have exhausted the memory it is time to clean up and remove triplets that are of no interest. Conceptually we

form linear combinations of the vectors,  $\tilde{\mathbf{u}} = \hat{\mathbf{U}}\mathbf{k}_u$  and  $\tilde{\mathbf{v}} = \hat{\mathbf{V}}\mathbf{k}_v$  and then start from scratch with these vectors as starting-vectors.

In principle it should be most flexible to allow  $\mathbf{k}_u$  and  $\mathbf{k}_v$  to be different. However, in bidiagonalization only one vector is used, and the theory is almost invariably developed by rewriting eq. (7.3) in the following form:

$$\mathbf{C}\mathbf{Q} = \begin{bmatrix} & \mathbf{A} \\ \mathbf{A}^H & \end{bmatrix} \begin{bmatrix} \hat{\mathbf{U}} \\ \hat{\mathbf{V}} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{U}} \\ \hat{\mathbf{V}} \end{bmatrix} \hat{\mathbf{B}} + \begin{bmatrix} \mathbf{E}_U \\ \mathbf{E}_V \end{bmatrix}$$

In exact arithmetic  $\text{range}(\mathbf{Q})$  would form a Krylov-subspace of  $\mathbf{C}$  with  $\mathbf{q} = \begin{bmatrix} \hat{\mathbf{u}} \\ \hat{\mathbf{v}} \end{bmatrix}$  as starting-vector. We could therefore write the new starting-vector as  $\tilde{\mathbf{q}} = c_0\mathbf{q} + c_1\mathbf{C}\mathbf{q} + \dots + c_k\mathbf{C}^k\mathbf{q}$ . In other words, the restarting vector is found by multiplying a polynomial in  $\mathbf{C}$  with the original starting-vector. By factoring this polynomial we may therefore write:

$$\tilde{\mathbf{q}} = \left( \prod_i (\mathbf{C} - \mu_i \mathbf{I}) \right) \mathbf{q}$$

In 1994 Åke Björck, Eric Grimme and Pail Van Dooren published a method for restarting the Lanczos bidiagonalization algorithm implicitly [9]. Their method was an adaption of a scheme by Danny C. Sorensen for restarting the Arnoldi method.

The general idea in implicit restarting is that all matrices are transformed to the state they would have had if the new starting vector had been used, but without going through all the steps again. The zeros of the polynomial are used, not the coefficients, in a series of shifted QR-decomposition. We refer to [50] for details.

### The exact shift strategy

A good introduction to different restarting strategies found in real software is [49]. Here we merely recognize that the most successful restarting strategy is the exact

#### Algorithm 7.7: Reorthogonalization

*Given an NPSVD with well-conditioned singular matrices, this function restore column-orthogonality to full machine precision. The resulting coupling-matrix is full in general.*

```

function [ $\tilde{\mathbf{U}}, \tilde{\mathbf{V}}, \tilde{\mathbf{B}}, \tilde{\mathbf{s}}_u, \tilde{\mathbf{s}}_v, \tilde{\mathbf{f}}_u, \tilde{\mathbf{f}}_v$ ] = orthogonalize( $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \mathbf{s}_u, \mathbf{s}_v, \mathbf{f}_u, \mathbf{f}_v$ )
    [ $\tilde{\mathbf{U}}, \mathbf{R}_U$ ] = qr( $\hat{\mathbf{U}}, 0$ );    [ $\tilde{\mathbf{V}}, \mathbf{R}_V$ ] = qr( $\hat{\mathbf{V}}, 0$ );           Determine QR-factorizations
     $\mathbf{x}_u = \tilde{\mathbf{U}}^H * \mathbf{s}_u$ ;     $\tilde{\mathbf{s}}_u = \mathbf{s}_u - \tilde{\mathbf{U}} * \mathbf{x}_u$ ;           Update  $\mathbf{s}_u$  by eq. (7.19)
     $\mathbf{x}_v = \tilde{\mathbf{V}}^H * \mathbf{s}_v$ ;     $\tilde{\mathbf{s}}_v = \mathbf{s}_v - \tilde{\mathbf{V}} * \mathbf{x}_v$ ;           Update  $\mathbf{s}_v$ 
     $l_u = \text{size}(\hat{\mathbf{B}}, 1) - \text{length}(\mathbf{f}_u)$ ;     $\mathbf{z}_u = \text{zeros}(l_u, 1)$ ;           Form implicit zeros in  $\mathbf{f}_u$ 
     $l_v = \text{size}(\hat{\mathbf{B}}, 2) - \text{length}(\mathbf{f}_v)$ ;     $\mathbf{z}_v = \text{zeros}(l_v, 1)$ ;
     $\tilde{\mathbf{f}}_u = ((\mathbf{R}_V)^{-1})^H * [\mathbf{z}_u; \mathbf{f}_u]$ ;           Update  $\mathbf{f}_u$ , eq. (7.20)
     $\tilde{\mathbf{f}}_v = ((\mathbf{R}_U)^{-1})^H * [\mathbf{z}_v; \mathbf{f}_v]$ ;           Update  $\mathbf{f}_v$ 
    if 1,  $\tilde{\mathbf{B}} = \mathbf{R}_U * \hat{\mathbf{B}} * (\mathbf{R}_V)^{-1} + \mathbf{x}_u * \tilde{\mathbf{f}}_u^H$ ;           Update  $\tilde{\mathbf{B}}$  using eq. (7.17a)
    else,  $\tilde{\mathbf{B}} = ((\mathbf{R}_U)^{-1})^H * \hat{\mathbf{B}} * \mathbf{R}_V^H + \tilde{\mathbf{f}}_v * \mathbf{x}_v^H$ ;           Or use (7.17b)
    [ $\tilde{\mathbf{s}}_u, \alpha$ ] = unit( $\tilde{\mathbf{s}}_u$ );     $\tilde{\mathbf{f}}_u = \tilde{\mathbf{f}}_u * \alpha$ ;           Normalize
    [ $\tilde{\mathbf{s}}_v, \alpha$ ] = unit( $\tilde{\mathbf{s}}_v$ );     $\tilde{\mathbf{f}}_v = \tilde{\mathbf{f}}_v * \alpha$ ;           Normalize

```

shift strategy used in ARPACK. Other strategies, such as shifting at Leja-points or roots of a Chebyshev polynomial, may also be useful [13].

The exact shift strategy has the following appealing interpretation when run in exact arithmetic [50, p. 61–62]:

1. Diagonalize the coupling-matrix,  $\hat{\mathbf{B}}$ , as described in sec. 7.2.
2. Sort the diagonal into disjoint sets of “wanted” and “unwanted” values.
3. Form a linear combination,  $\tilde{\mathbf{q}} = \tilde{\mathbf{Q}}\mathbf{k}$ , of the columns in the wanted set.
4. Begin all over with  $\tilde{\mathbf{q}}$  as the starting-vector, and update once for each entry in the wanted set.

The spectrum of the restarted coupling-matrix now consists of the wanted set.

## 7.8 Restarting the NPSVD

If  $\hat{\mathbf{B}}$  is diagonal, then deleting a column from  $\hat{\mathbf{U}}$ ,  $\hat{\mathbf{V}}$ ,  $\mathbf{E}_U$ ,  $\mathbf{E}_V$ , and both column and row from  $\hat{\mathbf{B}}$  is still a valid NPSVD. Comparing with the description in the previous section, we see that deleting from a diagonal NPSVD is essentially equivalent to implicit restarting with the exact shift strategy.

The diagonalization itself was derived in sec. 7.2. As in the previous sections we extend this to the case where we are given residual approximations  $\mathbf{E}_U \approx \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H$  and  $\mathbf{E}_V \approx \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H$ . Adding and subtracting these approximations in eq. (7.6) leads to:

$$\begin{aligned}\tilde{\mathbf{E}}_U &= \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H \mathbf{V}_B + (\mathbf{E}_U - \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H) \mathbf{V}_B \\ \tilde{\mathbf{E}}_V &= \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H \mathbf{U}_B + (\mathbf{E}_V - \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H) \mathbf{U}_B\end{aligned}$$

Apart from giving equations for the transformed search-vectors we also see that no amplification of previous rounding errors occurs (measured in any unitarily invariant norm).

001010  
100001  
111100

### Algorithm

Instead of deleting columns of  $\tilde{\mathbf{U}}$  we may delete columns of  $\mathbf{U}_B$  before applying the product, as shown in algorithm 7.8. This is a significant improvement in large-scale applications.

One significant attribute is that the majority of work is expressed in the form of matrix-multiplications, which is available at peak-performance on modern computers. In fact, orthogonalization, alg. 7.7, and restart can be fused into a single efficient operation. This will be pursued in the applications in the coming chapters.

## 7.9 References

Krylov methods are very general, and the subject is treated in numerous places. Two excellent examples are books written by Trefethen and Bau [73] and Demmel [15].



### NPSVD Restart

Let  $\mathbf{E}_U \approx \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H$  and  $\mathbf{E}_V \approx \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H$  be rank-1 approximations, and  $\hat{\mathbf{B}} = \mathbf{U}_B \boldsymbol{\Sigma}_B \mathbf{V}_B^H$  an SVD. Then the restarted NPSVD is given by:

$$(7.24) \quad \tilde{\mathbf{U}} = \hat{\mathbf{U}} \mathbf{U}_B \quad \tilde{\mathbf{B}} = \boldsymbol{\Sigma}_B \quad \tilde{\mathbf{V}} = \hat{\mathbf{V}} \mathbf{V}_B$$

$$(7.25) \quad \tilde{\mathbf{s}}_u = \hat{\mathbf{s}}_u \quad \tilde{\mathbf{s}}_v = \hat{\mathbf{s}}_v$$

$$(7.26) \quad \tilde{\mathbf{f}}_u = \mathbf{V}_B^H \hat{\mathbf{f}}_u \quad \tilde{\mathbf{f}}_v = \mathbf{U}_B^H \hat{\mathbf{f}}_v$$

$$(7.27) \quad \tilde{\mathbf{E}}_U = \tilde{\mathbf{s}}_u \tilde{\mathbf{f}}_u^H + (\mathbf{E}_U - \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H) \mathbf{V}_B$$

$$(7.28) \quad \tilde{\mathbf{E}}_V = \tilde{\mathbf{s}}_v \tilde{\mathbf{f}}_v^H + (\mathbf{E}_V - \hat{\mathbf{s}}_v \hat{\mathbf{f}}_v^H) \mathbf{U}_B$$

Now split the singular values of  $\hat{\mathbf{B}}$  into a “wanted” and “unwanted” set, and delete the following entries that corresponds to the unwanted set:

- Entries in  $\tilde{\mathbf{s}}_u$  and  $\tilde{\mathbf{s}}_v$ ,
- Columns in  $\tilde{\mathbf{U}}$  and  $\tilde{\mathbf{V}}$ ,
- Columns and rows in  $\tilde{\mathbf{B}}$ ,

Note that the columns in  $\tilde{\mathbf{E}}_U$  and  $\tilde{\mathbf{E}}_V$  are deleted implicitly.

For the bidiagonalization, one of the most complete references is the paper where Paige and Saunders develops the iterative solver known as LSQR [55]. A thorough analysis of the effect of rounding errors in the case of partial reorthogonalization can be found in the PhD thesis by Rasmus Munk Larsen [46].

Implicit restarting was originally developed for the Arnoldi-iteration by Danny C. Sorensen [66]. The paper was expanded into a more explanatory technical report [67].

The restarting scheme is implemented in the highly successful software-package ARPACK written by Richard Bruno Lehoucq et. al. [50]. Those who are interested in restarting strategies in real software should also consult his survey in [49].

Exact restarting is the most successful strategy used in current software. However,

#### Algorithm 7.8: Restart NPSVD

*Example of a simple NPSVD restart, assuming the singular matrices have orthonormal columns (e.g. returned by alg. 7.7). The  $N$  approximate triplets with singular vectors nearest  $\alpha$  is chosen for the restart.*

```

function [ $\tilde{\mathbf{U}}, \tilde{\mathbf{V}}, \tilde{\mathbf{B}}, \tilde{\mathbf{s}}_u, \tilde{\mathbf{s}}_v, \tilde{\mathbf{f}}_u, \tilde{\mathbf{f}}_v$ ] = restart( $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \mathbf{s}_u, \mathbf{s}_v, \mathbf{f}_u, \mathbf{f}_v, \alpha, N$ )
     $\mathbf{f}_u = [\text{zeros}(\text{size}(\hat{\mathbf{B}}, 1) - \text{length}(\mathbf{f}_u), 1); \mathbf{f}_u];$            Prepend implicit zeros in  $\mathbf{f}_u$ 
     $\mathbf{f}_v = [\text{zeros}(\text{size}(\hat{\mathbf{B}}, 2) - \text{length}(\mathbf{f}_v), 1); \mathbf{f}_v];$        And for  $\mathbf{f}_v$ 
    [ $\mathbf{U}_B, \boldsymbol{\Sigma}_B, \mathbf{V}_B$ ] = svd( $\hat{\mathbf{B}}, 0$ );  $\boldsymbol{\sigma} = \text{diag}(\boldsymbol{\Sigma}_B)$ ;      Compute the SVD of  $\hat{\mathbf{B}}$ 
    [ $\emptyset, \text{inx}$ ] = sort(| $\boldsymbol{\sigma} - \alpha$ |);  $\text{inx} = \text{inx}(1 : N)$ ;          Find wanted set as entries closest to  $\alpha$ 
     $\tilde{\mathbf{f}}_u = \mathbf{V}_B^H * \mathbf{f}_u$ ;  $\tilde{\mathbf{f}}_u = \tilde{\mathbf{f}}_u(\text{inx})$ ;  $\tilde{\mathbf{s}}_u = \mathbf{s}_u$ ;          Update  $\tilde{\mathbf{E}}_U$ , search-vector is unmodified
     $\tilde{\mathbf{f}}_v = \mathbf{U}_B^H * \mathbf{f}_v$ ;  $\tilde{\mathbf{f}}_v = \tilde{\mathbf{f}}_v(\text{inx})$ ;  $\tilde{\mathbf{s}}_v = \mathbf{s}_v$ ;      Same for  $\tilde{\mathbf{E}}_V$ 
     $\tilde{\mathbf{B}} = \boldsymbol{\Sigma}_B(\text{inx}, \text{inx})$ ;                                  The coupling-matrix was given by the SVD
     $\tilde{\mathbf{U}} = \hat{\mathbf{U}} * \mathbf{U}_B(:, \text{inx})$ ;  $\tilde{\mathbf{V}} = \hat{\mathbf{V}} * \mathbf{V}_B(:, \text{inx})$ ;      Update remaining matrices
  
```

other schemes are possible, with Leja points being one of the most interesting [13].

Those who want to use Krylov methods without too much thinking should be warned that it is necessary to pay extreme attention to details in floating point arithmetics. Therefore, it is best to use canned routines written by experts. Many excellent routines are available, e.g. in MATLAB<sup>®</sup> or ARPACK.

Those who want to implement Krylov algorithms anyway may start with the excellent book by Claerbout [14], and the book of template algorithms by Barrett et. al. [4].

# Computing the SVD

*This chapter applies the framework laid out in chapter 7 to compute large-scale SVD problems. A practical implementation faces a number of issues that must be dealt with.*

*First, we give a general description of high-performance implementations and strategies. This is followed by the basic iteration that forms the core of the SVD computations. After dealing with several practical problems, such as triplet aliasing, we give several deflation strategies.*

*Finally, these results are combined into the sliding window strategy.*

## 8.1 On the meaning of “large-scale”

The term “large-scale” is a somewhat fluid concept and often depends on context. Sometimes the term is used for any problem whose dimensions are so large that direct methods are unpractical.

We will use the term for systems that are so large that only a fraction of the singular triplets will fit in main memory of the computer. If this “fraction” is small enough, the computational cost of several of the modules in the previous chapter are severely affected.

Consider a problem represented by a matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$ ,  $m \geq n$ , where only  $K$  triplets can be held in memory. Assume the SVD of the  $k \times k$  coupling-matrix,  $\hat{\mathbf{B}}$ , is needed at some point.

A general SVD algorithm runs in  $\mathcal{O}(k^3)$  time while the matrix-matrix multiplication  $\hat{\mathbf{U}}\mathbf{Q}$ ,  $\mathbf{Q} \in \mathbb{R}^{k \times k}$  runs in  $\mathcal{O}(mk^2)$  time. Although the order-factors for the two problems are very different, the multiplication still dominates the time spent if  $k/m \leq K/m$  is small enough.

If the problem is very large, the work spent by any  $\mathcal{O}(k^3)$  algorithm is negligible compared to operations performed on the singular matrices. Two direct consequences of this should be kept in mind:

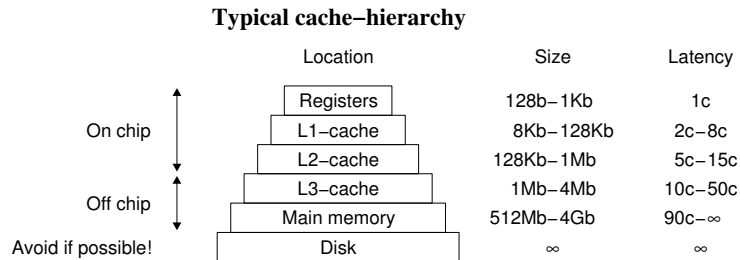
1. it is essential to optimize the work performed on the singular matrices,
2. the computational cost is not dominated by work on  $k \times k$  matrices.

Generally, it is sound advice to exploit the structure of a matrix. In contrast, the reorthogonalization discussed in sec. 7.6, and implemented in alg. 7.7 p. 91, destroys the structure of  $\hat{\mathbf{B}}$ . However, the extra cost of working on  $\hat{\mathbf{B}}$  is insignificant compared to the advantage gained by collecting the work on the singular matrices.

## 8.2 The memory hierarchy

The memory system of a modern computer is implemented in a number of layers as shown in fig. 8.1. In order to perform a computation rapidly, it is important to utilize the small amount of fast memory available.

**Figure 8.1:**  
Memory hierarchy of a typical modern computer. The small amount of fast memory must be utilized to achieve high performance.



From a practical point of view, this is achieved by *blocking* the operations. In practice, this means we should perform as much work as possible in highly optimized kernels. In the current context, these kernels come from two sources.

- **BLAS** (Basic Linear Algebra Subprograms). This is a standardized interface to basic operations. To achieve high performance, the library must be tuned to the computer architecture. A high-quality implementation depends on intricate details of the cache-implementations. An apparently simple routine, such as matrix-matrix multiplication, may use different methods depending on whether a matrix has a prime number of rows or is a power of two.
- **LAPACK** (Linear Algebra Package). The de-facto software package for small dense problem in linear algebra. It includes routines for a number of factorizations and solving basic problems. The package is built on BLAS, and is optimized for computers with deep cache hierarchies.

The BLAS-library is divided into level 1–3, which can roughly be characterized in the following way [15].

- **Level 1** Performs  $\mathcal{O}(k)$  operations. Examples are inner products, vector-scaling, and vector-norms.
- **Level 2** Performs  $\mathcal{O}(k^2)$  operations. Examples are matrix-vector products, solving a triangular system, and adding a rank-one update to a matrix.
- **Level 3** Performs  $\mathcal{O}(k^3)$  operations. Generally, these operations are various incarnations of matrix-matrix products.

The level-3 BLAS is important because the amount of computation is large compared to the data that must be accessed through memory. This makes it possible to exploit the speed of current computers, which is impossible with level-1 and level-2 operations.

Generally, it is wise to use level-3 operations if possible. CPU vendors, such as Intel and IBM, provide highly tuned libraries and using them can easily increase

the speed of an algorithm by a factor of 5–10, depending on the age of the computer. More importantly, this figure is expected to increase at an exponential rate at least until 2016 [63].

Finally, a procedure based on level-3 operations has the ability to scale to several processors for very large problems. Again this is only advantageous if the amount of processing is large compared to the data. This makes it reasonable to transmit the data over a slow link, such as a network, an internal cross-bar switch, or a cache residing on a different CPU.

### 8.3 Left residual factorization

From now on we restrict our attention to algorithms that would keep the right residual  $\mathbf{E}_V$  matrix at zero in absence of rounding errors. In other words we set  $\mathbf{E}_V \approx \mathbf{0}$  and seek to minimize  $\mathbf{E}_U \approx \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H$ , hence the name.

It is convenient to write the residuals in the form  $\mathbf{E}_U = \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H + \mathbf{G}_U$  and  $\mathbf{E}_V = \mathbf{G}_V$ . We can then rewrite the NPSVD in the following form:

$$(8.1a) \quad \mathbf{A} \hat{\mathbf{V}} = \hat{\mathbf{U}} \hat{\mathbf{B}} + \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u^H + \mathbf{G}_U$$

$$(8.1b) \quad \mathbf{A}^H \hat{\mathbf{U}} = \hat{\mathbf{V}} \hat{\mathbf{B}}^H + \mathbf{G}_V$$

All rounding errors are represented in  $\mathbf{G}_U$  and  $\mathbf{G}_V$ . We shall refer to these matrices as *rounding residual matrices*, or just rounding residuals.

#### Lanczos and residual factorization

By our definition Lanczos bidiagonalization, described in sec. 7.5.2 p. 87, is a left residual factorization. The new term is introduced because the coupling matrix  $\hat{\mathbf{B}}$  will not generally be on bidiagonal form.

Ronald B. Morgan gave an elegant proof in [53] that an orthogonalized NPSVD where  $\mathbf{E}_U$  is rank one,  $\mathbf{E}_V$  is zero, and the singular values of  $\hat{\mathbf{B}}$  are distinct corresponds to a bidiagonalization with a certain starting vector. Thus, it does not matter what form the coupling matrix has.

Since the algorithms to be presented in this chapter are similar to the Lanczos algorithms, it might be appropriate to highlight some differences between the framework from the previous chapter and the Lanczos theory as it is developed traditionally.

**Accuracy** Lanczos algorithms have a striking ability to determine singular values accurately, in spite of rounding errors. This is quite surprising, since the inner loop is a recurrence using only the previous vector. The result is nevertheless fully established in sec. 7.3, where *all* rounding errors are taken into account. To the best knowledge of the author, this has not been done previously.

**Modularization** In more practical terms, the systematic modularization of the routines is good engineering practice. It allows the same code to be used in several different applications, and makes it easier to construct reliable software. Since all the modules were shown to be stable, it is straightforward to combine them into more complex algorithms and prove stability.

Efficiency	Computational efficiency is perhaps best illustrated by delayed orthogonalization. The bulk of work in this module is naturally formulated as matrix-matrix multiplications, which are level-3 operations. However, this advantage can only be gained by abandoning the requirement that the coupling-matrix remains bidiagonal.
General restart	Furthermore, the implicit restarting scheme has some advantages not found in previous methods. Apart from the fact that the work is mainly performed as level-3 operations, it also has the advantage that computing a converged triplet and removing it is done in the same operation. This fact will be exploited extensively in the algorithms discussed in this chapter.

### Alternatives

We reiterate that the advantages only applies to large-scale problems as described in sec. 8.1. While some optimization could possibly make the algorithms competitive for “medium-scale” problems (in terms of computation time), they are not suitable for small-scale problems such as unstructured dense systems.

When this project was initiated, there was no codes available that combined implicit restarting and partial reorthogonalization. In the meantime, Rasmus Munk Larsen released a Fortran package [47] which extends the MATLAB<sup>®</sup>-code from his PhD [46] with this ability.

It is possible that this approach can be used as an alternative to the full reorthogonalization described in chapter 7. For now, the numerical behavior of the approach has not yet been fully analyzed.

Finally, it is possible that an algorithm based on partial reorthogonalization without restart will suffice for a particular problem. However, although implicitly restarted methods may be slower in a particular case, their ability to scale may justify their use.

## 8.4 Accuracy of matrix-vector product

The algorithms are based on functions for computing the matrix-vector products  $\mathbf{A}\mathbf{v}$  and  $\mathbf{A}^H\mathbf{u}$ , where  $\mathbf{u}$  and  $\mathbf{v}$  are unit vectors. The numerical properties of the algorithms are affected by the accuracy of these functions, and we need to quantify this.

We will assume there is a constant,  $\tau$ , so the functions satisfies  $f_A(\mathbf{v}) = \mathbf{A}\mathbf{v} + r_A(\mathbf{v})$  and  $f_A^H(\mathbf{u}) = \mathbf{A}^H\mathbf{u} + r_A^H(\mathbf{u})$  where  $\|r_A(\mathbf{v})\|_2 \leq \tau$  and  $\|r_A^H(\mathbf{u})\|_2 \leq \tau$ .

This model is well suited for applications where the rounding errors generated by the functions are insensitive to the input. It implies that the rounding errors are consider to be of the same order irrespective of the norm of the output.

In many cases  $\tau = c\epsilon_M\|\mathbf{A}\|_2$ , where  $c$  is a modest factor. If it is impossible to analyze the accuracy of the routines, setting  $c$  to be a comfortably large constant may be the only practical way to proceed.

### Sparse matrices

For concreteness, we consider the case where  $\mathbf{A}$  is sparse, and the matrix-vector product is computed by inner products. It follows from sec. 1.8 that we can set

$\tau = \nu_l \|(|\mathbf{A}|)\|_2$ , where  $l$  is the maximum number of nonzero elements in any row or column of  $\mathbf{A}$ .

Since the 2-norm is not absolute, we can not simply use  $\|\mathbf{A}\|_2$  in this equation. Instead we may estimate the norm separately, or we can use the bound  $\|(|\mathbf{A}|)\|_2 \leq \sqrt{\|\mathbf{A}\|_1 \|\mathbf{A}\|_\infty}$  [27]. However, such a bound is usually pessimistic since the rounding errors tends to be normally distributed and reaching the bound becomes extremely improbable.

### Asymmetric accuracy

It is possible that one computational routine is more accurate than the adjoint. For instance, consider a sparse matrix with only a few non-zero elements in each row, but some relatively dense columns. We saw in sec. 1.8 that in this case the product  $\mathbf{A}\mathbf{v}$  will usually be computed more accurately than  $\mathbf{A}^H\mathbf{u}$ .

Another example introducing asymmetry by structure is preconditioning. A preconditioner is often a diagonal matrix which can be applied to high relative accuracy. It has the effect of scaling elements to roughly the same order, and all else being equal, this commonly leads to smaller rounding errors. However, the routine implementing the adjoint operator applies the preconditioner at the end of the computation, which does not improve the accuracy.

Asymmetry also arises for algorithmic reasons. Consider a pair of algorithms that accesses a sparse matrix with non-zero elements stored in column-order. The product  $\mathbf{A}^H\mathbf{u}$  can be computed element by element in a tight inner loop. A computer with extended-precision registers can often compute each component very accurately. In contrast, the product  $\mathbf{A}\mathbf{u}$  is computed by accumulating a vector in memory, and does not benefit much from extended precision.

### Direct measurement

A different approach is to measure  $\tau$  based on the functions themselves. If the routines are available in different precisions, it may be possible to estimate  $\tau$  by comparing the different variants.

In practice, it is more useful to estimate  $\tau$  from a single set of algorithms. This is also useful for checking that the functions  $f_A$  and  $f_A^H$  actually implements the adjoint of each other.

Given that the accuracy of the matrix-vector products may be asymmetric, and often have a component whose size depends on the norm of the output, it is by no means a trivial task to estimate  $\tau$  reliably. We shall not give a full analysis here, but mention that the routine `bbtauest` in `BBTools` works well in many cases.

## 8.5 Basic iteration

The modules derived chapter 7 are given in a general form. This chapter concerns how these modules may be pieced together in the case of left residual factorization.

The basic iteration, or base-iteration, is essentially started as alg. 7.5 p. 87. This algorithm lacks a stopping criterion; the details are worked out in appendix B and a simplified implementation is given in alg. B1 p. 146.

When the iteration completes, we may orthogonalize and restart (alg. 7.8 and 7.7). Although the routines are limited to an exact restarting strategy, it is still necessary to choose the “wanted” and “unwanted” sets described in sec. 7.7. This will be addressed in the following sections.

The restarted, i.e. reduced, NPSVD is then expanded again via the base-iteration, until the desired singular triplets have been extracted. However, when we get to reorthogonalize, only the updated part of the singular matrices needs to be re-orthogonalized.

To implement this, it is convenient to include reorthogonalization in the base iteration. The QR-decompositions in alg. 7.7 can be computed efficiently by computing  $\mathbf{R}_U$  and  $\mathbf{R}_V$  as the Cholesky-factorizations  $\mathbf{R}_U^H \mathbf{R}_U = \hat{\mathbf{U}}^H \hat{\mathbf{U}}$  and  $\mathbf{R}_V^H \mathbf{R}_V = \hat{\mathbf{V}}^H \hat{\mathbf{V}}$ . The singular matrices may then be updated by post-multiplying by  $\mathbf{R}_U^{-1}$  and  $\mathbf{R}_V^{-1}$ .

This is not generally a stable approach, but is a viable option in this case where the condition numbers of  $\mathbf{R}_U$  and  $\mathbf{R}_V$  are kept under strict control. Apart from being a fast way to compute the QR-factorization, it also means that we can update  $\hat{\mathbf{B}}$  before  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$ .

However, when we know  $\hat{\mathbf{B}}$  we may diagonalize it and update  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  as a single matrix-matrix multiplication. Combining the basic updates, reorthogonalization, and diagonalization into the base-iteration reduces the work on the singular matrices to the following operations:

- Update  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  from, say,  $k_0$  to  $k$  columns (alg. B1).
- Compute columns<sup>1</sup>  $k_0 + 1$  to  $k$  of the matrices  $\hat{\mathbf{U}}^H \hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}^H \hat{\mathbf{V}}$ .
- Fuse reorthogonalization and diagonalization to update of  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  with a single matrix-matrix multiplication for each.

As discussed in sec. 8.1, the bulk of work in large-scale problems is performed in work involving the large matrices. In this case these are the singular matrices,  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$ , and the operator  $\mathbf{A}$ .

The fused procedure above updates the singular matrices mainly with level-3 operations. The expansion itself requires two matrix-vector products, with  $\mathbf{A}$  and  $\mathbf{A}^H$ , to add a column to each singular matrix. It is difficult to imagine this can be improved much further.

For details about the fused implementation we refer to [34], which also includes benchmarks of the process.

## 8.6 Accuracy and aliasing

When the base-iteration completes, we have an NPSVD on the form given by eq. (8.1) where the coupling matrix is diagonal,  $\hat{\mathbf{B}} = \text{diag}(\hat{\sigma})$ , and the singular matrices have columns that are numerically orthonormal.

The rounding residual matrices,  $\mathbf{G}_U$  and  $\mathbf{G}_V$ , are unknown, but their norms are on the order of  $\tau$ . They may generally be ignored in the restarting strategy; their role served in the analysis of the base iteration.

<sup>1</sup>This also gives the columns, since  $\mathbf{X}^H \mathbf{X}$  is Hermitian.



Once we have the ortho-diagonal NPSVD, we also have estimates of the singular triplets of  $\mathbf{A}$ . Ignoring the residual matrices, the quality of the triplets are determined by  $\hat{\mathbf{s}}_{\mathbf{u}}$ . Specifically, corollary C18 p. 155 states that  $\mathbf{A}$  has a singular value,  $\sigma$ , in the range  $|\hat{\sigma}|_i - |\hat{\mathbf{s}}_{\mathbf{u}}|_i \leq \sigma \leq |\hat{\sigma}|_i + |\hat{\mathbf{s}}_{\mathbf{u}}|_i$ .

Thus, if triplet  $i$  satisfies the  $|\hat{\mathbf{s}}_{\mathbf{u}}|_i \leq \tau$ , then it has converged to the accuracy provided by the matrix-vector product routines. After convergence, a triplet may be removed to make space for new iterates.

### Triplet aliasing

The fallacy of this is that there are many other triplets that may contain useful information, and it would be wasteful at best to throw these away. An in depth discussion of this point is given in sec. C7 on p. 153, and the result is that we should also include all triplets that may *alias* triplets in the wanted set.

In short, an aliasing triplet is a triplet that may be rich in components corresponding to triplets already chosen for the restart. If we do not include these triplets for the restart, the convergence may be slow because the algorithm is required to “re-discover” these directions. Even worse, we may throw away a triplet with a large components corresponding to a singular value that is smaller than those found already. This is not acceptable, and leads to algorithms that may not converge at all.

Fortunately, it is not hard to determine which triplets that may alias those in the wanted set. All we need is a bound on the singular values corresponding to each triplet. This is supplied by theorem C17 and implemented in alg. 8.1.

### Combating rounding errors

Rounding errors are the bane of efficiency for iterative algorithms. While detailed analysis is necessary to get numerical algorithms stable, an element of art remains as defined by Donald E. Knuth [56]:

Science is what we understand well enough to explain to a computer.  
Art is everything else we do.

After a restart, the norm of  $\mathbf{E}_{\mathbf{U}}$  may be very small. This is particularly true if the restart discards some triplets with large errors. The result is that the norm of  $\mathbf{E}_{\mathbf{U}} \approx \hat{\mathbf{s}}_{\mathbf{u}} \hat{\mathbf{f}}_{\mathbf{u}}$  may be reduced by a significant factor during the restart.

**Algorithm 8.1:** Expand a set of triplets to include aliasing triplets.

*Given an NPSVD and error-bounds, this routine find intervals for singular values of each triplets. It returns the triplets that aliases those in the wanted set.*

---

```

function inx = widen(inx,  $\sigma$ ,  $\hat{\mathbf{f}}_{\mathbf{u}}$ )
   $\sigma_{\text{lo}} = \sigma + \text{abs}(\hat{\mathbf{f}}_{\mathbf{u}})$ ;                                     Maximum aliased singular value
   $\sigma_{\text{hi}} = \text{max}(\sigma - \text{abs}(\hat{\mathbf{f}}_{\mathbf{u}}), 0)$ ;                   Minimum aliased singular value
  Max = max( $\sigma_{\text{lo}}$ (inx));   Min = min( $\sigma_{\text{hi}}$ (inx));       Find extreme bounds for requested triplets
  inx = find(( $\sigma_{\text{hi}} \leq \text{Max}$ )  $\wedge$  ( $\sigma_{\text{lo}} \geq \text{Min}$ ));     Find aliasing triplets

```

---

Although it is known that  $\|\mathbf{E}_U - \hat{\mathbf{s}}_u \hat{\mathbf{f}}_u\|$  is small at the point of the restart, any component in  $\hat{\mathbf{s}}_u$  corresponding to singular triplets of  $\mathbf{A}$  with large singular values, will be picked up immediately.

One way to improve this is to improve the rank-one approximation by taking a power-step of  $\mathbf{E}_U = \mathbf{A}\hat{\mathbf{V}} - \hat{\mathbf{U}}\hat{\mathbf{B}}$ . We note that this improvement is empirical, and does not change any error-bounds, and is redundant in the absence of rounding errors.

## 8.7 Singular value decomposition

We are now ready to attack the problem of computing the SVD of a large-scale matrix  $\mathbf{A}$ . For this we will assume that  $\mathbf{A} \in \mathbb{C}^{m \times n}$  where  $m \geq n$ . Otherwise we may instead compute the SVD of  $\mathbf{A}^H$ .

Our problem is to compute as many triplets, corresponding to the largest singular values, with a restricted amount of memory. We note that computing the eigen-decomposition  $\hat{\mathbf{V}}\hat{\Sigma}^2\hat{\mathbf{V}} = \mathbf{A}^H\mathbf{A}$  only needs to store vectors of length  $n$ , which may be a very substantial saving.

This is a viable approach for relatively well-conditioned systems. For problems that does not need singular values below about  $\sqrt{\epsilon_M}$ , and have modest requirements about accuracy, this can be done efficiently using ARPACK [50].

As an example, assume that  $n = 10000$ , whereas  $m \gg n$  is large. If the matrix  $\hat{\mathbf{V}}$  is stored as IEEE double-precision, then it can be stored in roughly 1.6 GB of memory. We should therefore be able to handle such a system on an off-the-shelf desktop computer, regardless of  $m$ .

Our goal is to compute the SVD as accurately as possible, without using more resources than the scheme above. We also want to compute triplets beyond those that will fit in memory.

The first thing to notice is that the first few singular triplets can be computed efficiently by restarting. This requires storage of both left and right singular vectors, and after convergence they will be returned to the user for whatever problem the SVD is used to solve.

Since we do not have storage to keep these triplets in memory they must be removed from further processing. However, these triplet will eventually reappear because of rounding errors. Even in the absence of rounding errors they will reappear, unless the removed triplets were exact. This would rarely be the case in any iterative scheme.

## 8.8 Explicit deflation

One solution is to deflate the triplets. Basically, this amount to continue processing on the matrix  $\mathbf{A}' = \mathbf{A} - \mathbf{Q}_U\mathbf{S}\mathbf{Q}_V^H$ , where  $\mathbf{Q}_U$ ,  $\mathbf{S}$ , and  $\mathbf{Q}_V$  represents converged triplets:

$$\begin{aligned}\mathbf{A}\mathbf{Q}_V &= \mathbf{Q}_U\mathbf{S} + \mathbf{C}_U \\ \mathbf{A}^H\mathbf{Q}_U &= \mathbf{Q}_V\mathbf{S}^H + \mathbf{C}_V\end{aligned}$$

According to the Schmidt-Mirsky theorem (see p. 150) this minimizes the norm of  $\mathbf{A}'$ , if the singular triplets are accurate. We shall refer to this as *two-sided deflation*, since both the left and right singular triplets are explicitly kept orthogonal to converged triplets.

Unfortunately, two-sided deflation stores all the converged triplets and does not save memory. Nevertheless, the scheme remains useful as a component of a larger strategy where only a few components are stored.

As an alternative we may turn to *one-sided deflation*:

$$\begin{aligned}
 \mathbf{A}' &= \mathbf{A}(\mathbf{I} - \mathbf{Q}_V \mathbf{Q}_V^H) \\
 &= \mathbf{A} - (\mathbf{Q}_U \mathbf{S} + \mathbf{C}_U) \mathbf{Q}_V^H \\
 &= \mathbf{A} - \mathbf{Q}_U \mathbf{S} \mathbf{Q}_V^H - \mathbf{C}_U \mathbf{Q}_V^H
 \end{aligned}
 \tag{8.2}$$

This is almost as good, since  $\|\mathbf{C}_U\|$  is less than the tolerance we require for the singular triplets.

It is necessary to ensure that changing the matrix does not disturb the restart. We note that before restarting all triplets have been orthogonalized against each other. It follows that all non-discarded triplets *could* have been computed using  $\mathbf{A}'$  instead of  $\mathbf{A}$ . It is therefore stable to move converged vectors into  $\mathbf{Q}_V$  before restarting.

However, to keep the updates consistent we are required to compute all further matrix-vector products using  $\mathbf{A}'$  instead of  $\mathbf{A}$ . Otherwise the NPSVD will be inconsistent, and unpredictable results may occur. While this represents a significant amount of work in level-2 operations, it is acceptable if  $n \ll m$ .

The second point is that we must ensure the accuracy of the remaining triplets. So if  $\hat{\mathbf{U}}$ ,  $\hat{\mathbf{B}}$ , and  $\hat{\mathbf{V}}$  is an NPSVD of  $\mathbf{A}'$  we get:

$$\begin{aligned}
 \mathbf{A}(\mathbf{I} - \mathbf{Q}_V \mathbf{Q}_V^H) \hat{\mathbf{V}} &= \hat{\mathbf{U}} \hat{\mathbf{B}} + \mathbf{E}_U \\
 (\mathbf{I} - \mathbf{Q}_V \mathbf{Q}_V^H) \mathbf{A}^H \hat{\mathbf{U}} &= \hat{\mathbf{V}} \hat{\mathbf{B}}^H + \mathbf{E}_V
 \end{aligned}$$

This implies:

$$\begin{aligned}
 \mathbf{A} \hat{\mathbf{V}} &= \hat{\mathbf{U}} \hat{\mathbf{B}} + \mathbf{E}_U + \mathbf{A} \mathbf{Q}_V \mathbf{Q}_V^H \hat{\mathbf{V}} \\
 &= \hat{\mathbf{U}} \hat{\mathbf{B}} + \mathbf{E}_U + (\mathbf{Q}_U \mathbf{S} + \mathbf{C}_U) (\mathbf{Q}_V^H \hat{\mathbf{V}})
 \end{aligned}
 \tag{8.3a}$$

$$\begin{aligned}
 \mathbf{A}^H \hat{\mathbf{U}} &= \hat{\mathbf{V}} \hat{\mathbf{B}}^H + \mathbf{E}_V + \mathbf{Q}_V \mathbf{Q}_V^H \mathbf{A}^H \hat{\mathbf{U}} \\
 &= \hat{\mathbf{V}} \hat{\mathbf{B}}^H + \mathbf{E}_V + \mathbf{Q}_V (\mathbf{S}^H \mathbf{Q}_U^H + \mathbf{C}_U^H) \hat{\mathbf{U}} \\
 &= \hat{\mathbf{V}} \hat{\mathbf{B}}^H + \mathbf{E}_V + \mathbf{Q}_V \mathbf{C}_U^H \hat{\mathbf{U}} + \mathbf{Q}_V \mathbf{S}^H (\mathbf{Q}_U^H \hat{\mathbf{U}})
 \end{aligned}
 \tag{8.3b}$$

Since we orthogonalize all vectors in  $\hat{\mathbf{V}}$  explicitly, we know that  $\|\mathbf{Q}_V^H \hat{\mathbf{V}}\|$  is negligible. However, it takes quite a bit of work to get a rigorous analysis of the complete case. The theoretical underpinnings of one-sided deflation is developed in appendix C, and we refer in particular to the orthogonality theorem on p. 152.

From a practical point of view, the result is that one-sided deflation is suitable for the following purposes:

- Accurate singular values (i.e. no singular vectors).
- Low-rank approximations.

- Any problem where orthogonality of the vectors in the long-space is unimportant.

## 8.9 Implicit deflation

Another option is to implicit deflation, which has far more potential for memory conservation. This is based on theorem C14 p. 152, which states that numerical singular triplets are numerical orthogonal, provided they are accurate and have different singular values.

In principle, we may keep a list of the singular values returned to the user, and if we find a new triplet that correspond to the same singular triplet, then we simply discard it without returning it again. This is attractive, because it keeps the storage requirement independent of *both*  $m$  and  $n$ , and does not introduce any additional rounding errors. However, it suffers from two drawbacks: some triplet may be missed and a very large number of matrix-vector products with  $\mathbf{A}$  may be required.

The first issue can be dealt with by observing that triplets converge roughly in decreasing order of singular values. However, if there are clustered triplets it takes many iterations before a single triplet will split and resolve these individually. An elaborate explanation of this phenomena can be found in [73].

We may deal with this by explicitly deflating a number of the most recently converged triplets. This makes it very unlikely that any triplet will be glossed over, but still avoid the need to keep all previous triplets in storage.

The other issue is more problematic, and arises from the fact that rounding errors quickly generate components in directions corresponding to large singular values as discussed in sec. 8.6.2. Unfortunately, there seems to be no other cure than either deflating explicitly or accept a large number of matrix-vector products.

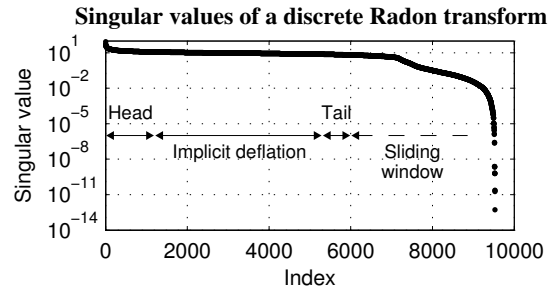
However, there are important applications where implicit deflation is useful. In many applications the singular values of  $\mathbf{A}$  reaches a plateau. This is typical for applications that depends on low-rank approximations, e.g. latent semantic indexing [6]. It is often interesting to know both *whether* the singular values eventually starts to drop, and if they do, where. Implicit deflation can answer this question in an inexpensive way.

We may then use implicit deflation as long as the largest implicitly deflated singular value is not too large compared to the remaining triplets. Theoretically, there is no bound but in practice the number of matrix-vector products becomes excessive if the range of implicit deflation is about a factor of 10.

## 8.10 Sliding window

The procedures described above can be combined into an aggressive restarting strategy, which we shall refer to as “sliding window”. The idea is sketched in fig. 8.2: explicit deflation (one- or two-sided) is used to keep large singular triplets from disturbing the computation (the head). Another set (the tail) is used to create a gap of singular values. This allows a large number of triplets to be deflated implicitly: any triplet with a singular value that exceeds the smallest implicitly

**Figure 8.2:**  
 Sketch of a step  
 where a sliding  
 window is used to  
 compute the SVD  
 of a discrete Radon  
 transform.



singular value is removed from the restart (unless it aliases triplets in the sliding window).

The best number of triplets to allocate for the head and tail depends on the problem at hand. The default behavior for the implementation in BBTools is to use a tail for all requested singular triplets. Tuning these to the problem at hand may significantly improve performance, but the other approach is very safe provided the system has enough memory.

## 8.11 References

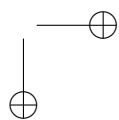
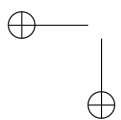
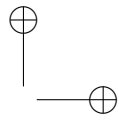
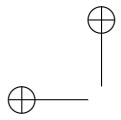
Most of the material on computer architecture can be found in text-books on numerical linear algebra, e.g. [15] and [69]. Those who want to implement their own BLAS may appreciate the difficulties by looking at ATLAS (Automatically Tuned Linear Algebra Software) [80].

The de-facto software package for numerical linear algebra with dense systems is LAPACK [2]. There is also a more modern variant, ScaLAPACK, which is intended for parallel computations [10].

Iterative computation of the SVD is supported by the following packages:

- ARPACK [50], described on p. 93, is written in Fortran. It forms the basis of the MATLAB<sup>®</sup>-functions `eigs` and `svds`.
- SVDPACK [7], by Michael W. Berry, contains several SVD-routines for large-scale computations. The routines, described in [6], are written in Fortran.
- PROPACK [47] includes Lanczos bidiagonalization with partial reorthogonalization. The original report [46] did not contain implicit restarting, but the software have now been expanded. The package exists for both Fortran and Matlab.

It should be noted that PROPACK only includes the bidiagonalization procedure. As should be evident from the contents of this chapter, using it for computing SVDs may require some additional work.



# Solving linear systems

This chapter builds on the two previous chapters to produce algorithms that solve linear systems. After treating the unregularized case, we show how regularized solutions, based on filter-factors, can be estimated efficiently.

The chapter finishes with a description of a practical hybrid solver that is suitable for solving tomographic problems.

## 9.1 Prelude to iterative solvers

The goal of a linear solver is to determine a vector,  $\hat{\mathbf{x}}$ , so that  $\mathbf{A}\hat{\mathbf{x}} \approx \mathbf{b}$  for a given  $\mathbf{b}$ . An iterative solver works by approximately solving a system for a residual:

$$(9.1) \quad \mathbf{A}\hat{\mathbf{x}}_i \approx \mathbf{r}_i$$

This is initialized with  $\mathbf{r}_0 = \mathbf{b}$ , i.e. the original problem. At step  $i$  an approximate solution,  $\mathbf{A}\hat{\mathbf{y}}_i \approx \mathbf{r}_i$  is determined. Generally, this estimate may be quite poor, so we update the equation system:

$$(9.2a) \quad \mathbf{r}_{i+1} = \mathbf{r}_i - \mathbf{A}\hat{\mathbf{y}}_i$$

$$(9.2b) \quad \hat{\mathbf{x}}_i = \sum_{k=0}^i \hat{\mathbf{y}}_k$$

Applying eq. (9.2) repeatedly leads to the expression:

$$\begin{aligned} \mathbf{r}_0 &= \hat{\mathbf{r}}_1 + \mathbf{A}\mathbf{y}_0 \\ &= \hat{\mathbf{r}}_2 + \mathbf{A}\mathbf{y}_1 + \mathbf{A}\mathbf{y}_0 \\ &= \hat{\mathbf{r}}_N + \sum_{k=0}^{N-1} \mathbf{A}\mathbf{y}_k \\ &= \hat{\mathbf{r}}_N + \mathbf{A}\hat{\mathbf{x}}_{N-1} \end{aligned}$$

If  $\|\mathbf{r}_N\|$  is small, then we have determined an accurate solution to the system  $\mathbf{A}\hat{\mathbf{x}} = \mathbf{r}_0 = \mathbf{b}$  as desired. This technique is sometimes used to combat rounding errors for direct solvers, where it is known as *iterative refinement* [69].

The strength is that the approximate solution may be quite inaccurate, as long as the residual norm is decreasing. To emphasize this, alg. 9.1 implements this by using a random vector to approximate the solution. This is allowed as long as the residual norm decreases in each step, which can be guaranteed by scaling.

Although simple, alg. 9.1 has most of the components necessary for an iterative solver, including a stopping-criterion that suffices if  $\mathbf{A}\mathbf{x} = \mathbf{b}$  has a solution.

## 9.2 Bidiagonalizing solver

The Lanczos bidiagonalization procedure, alg. 7.5 p. 87, is an excellent base for an iterative solver. We shall use the left residual factorization described in sec. 8.3. More precisely, the routines described in this chapter are based on the basic iteration described in sec. 8.5. Therefore, the effect of rounding-errors will not be addressed here.

Ignoring the rounding residual matrices, the basic iteration result in an NPSVD on the following form:

$$(9.3a) \quad \mathbf{A}\hat{\mathbf{V}} = \hat{\mathbf{U}}\hat{\mathbf{B}} + \hat{\mathbf{s}}_{\mathbf{u}}\hat{\mathbf{f}}_{\mathbf{u}}^{\mathbf{H}}$$

$$(9.3b) \quad \mathbf{A}^{\mathbf{H}}\hat{\mathbf{U}} = \hat{\mathbf{V}}\hat{\mathbf{B}}^{\mathbf{H}}$$

Here the coupling-matrix is diagonal,  $\hat{\mathbf{B}} = \text{diag}(\hat{\boldsymbol{\sigma}}) = \text{diag}([\hat{\sigma}_1, \dots, \hat{\sigma}_k])$ , and the singular matrices,  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$ , have orthonormal columns to machine precision.

The trick is to create the NPSVD using  $\mathbf{b}$  as the starting-vector (e.g. using  $\hat{\mathbf{u}} = \mathbf{b} = \mathbf{r}_0$  in alg. 7.5 p. 87). After the update is complete, we have the following relation:

$$(9.4) \quad \mathbf{r}_0 \in \text{span}([\hat{\mathbf{U}}, \hat{\mathbf{s}}_{\mathbf{u}}])$$

On the other hand, we can use the NPSVD to determine the result of any approximate solution on the form:

$$(9.5) \quad \hat{\mathbf{y}}_0 = \hat{\mathbf{V}}\hat{\mathbf{z}}_0$$

Inserting in eq. (9.3a):

$$(9.6) \quad \mathbf{A}\hat{\mathbf{y}}_0 = (\hat{\mathbf{U}}\hat{\mathbf{B}} + \hat{\mathbf{s}}_{\mathbf{u}}\hat{\mathbf{f}}_{\mathbf{u}}^{\mathbf{H}})\hat{\mathbf{z}}_0$$

From this there are two main ways to select  $\hat{\mathbf{y}}_0$ .

### Optimal (MINRES)

The MINRES algorithm (minimal residual) [73] is an algorithm which use Lanczos tridiagonalization on a Hermitian matrix  $\mathbf{A} = \mathbf{A}^{\mathbf{H}}$ . It has been generalized to the non-Hermitian case, where it is known as GMRES (generalized minimal residual).

**Algorithm 9.1:** Solve  $\mathbf{A}\mathbf{x} = \mathbf{b}$  iteratively using random approximations.

*This program is the extreme case of a bad approximate solution. If the system has a solution, the algorithm slowly converges to a solution with arbitrarily small residual norm (ignoring rounding errors).*

```

function [ $\hat{\mathbf{x}}, \mathbf{r}$ ] = randsolve( $\mathbf{A}, \mathbf{b}, \text{tol}$ )
    [ $m, n$ ] = size( $\mathbf{A}$ );  $\hat{\mathbf{x}} = \text{zeros}(n, 1)$ ;  $\mathbf{r} = \mathbf{b}$ ;           Setup system
     $T = \|\mathbf{b}\|_2 * \text{tol}$ ;                                       Determine stop criterium
    while  $\|\mathbf{r}\|_2 > T$ 
         $\hat{\mathbf{y}} = \text{randn}(n, 1)$ ;  $\mathbf{p} = \mathbf{A} * \hat{\mathbf{y}}$ ;                 Approximate solution by a random (!!!) vector
         $\alpha = (\mathbf{p}^{\mathbf{H}} * \mathbf{r}) / (\mathbf{p}^{\mathbf{H}} * \mathbf{p})$ ;  $\hat{\mathbf{y}} = \hat{\mathbf{y}} * \alpha$ ;  $\mathbf{p} = \mathbf{p} * \alpha$ ; Rescale to minimize residual
         $\mathbf{r} = \mathbf{r} - \mathbf{p}$ ;  $\hat{\mathbf{x}} = \hat{\mathbf{x}} + \hat{\mathbf{y}}$ ;                   Update residual and solution
    endwhile

```



It is based on the idea that  $\hat{\mathbf{y}}_0$  should be chosen to minimize the residual norm  $\|\mathbf{r}_0\|$ . From a practical point of view, this can be accomplished by solving the normal-equations on eq. (9.6) using Woodbury's formula [69]. While this is generally unstable, it is stable and efficient if the NPSVD is in ortho-diagonal form.

In exact arithmetic, this procedure is equivalent to MINRES on the normal equations:

$$(9.7) \quad \mathbf{A}^H \mathbf{A} \hat{\mathbf{y}}_0 = \mathbf{A}^H \mathbf{r}_0$$

The main problem with MINRES is that it requires an *explicit restart*: once the new residual is computed, the NPSVD can not be reused since eq. 9.4 fails to hold. Instead, a new NPSVD must be created starting from the new residual.

### Galerkin (Conjugate Gradient)

The deficiency of a MINRES restart leads us to look for a solution that allows an implicit restart of the NPSVD. This can be accomplished via the Galerkin criteria:  $\hat{\mathbf{y}}_0$  is chosen so  $\hat{\mathbf{r}}_1$  is orthogonal to the triplets removed during the restart. This is equivalent to CGNR: the conjugate gradient on the normal-equations in eq. (9.7).

More specifically, consider the case where we want to remove triplet  $i$  from the NPSVD. For this to work, the relation in eq. (9.4) must still hold after the restart. To this end consider the following quantity:

$$(9.8) \quad \begin{aligned} \text{By eq. (9.2a)} \quad \mathbf{u}_i^H \mathbf{r}_1 &= \mathbf{u}_i^H (\mathbf{r}_0 - \mathbf{A} \hat{\mathbf{y}}_0) \\ \text{By eq. (9.6)} \quad &= \mathbf{u}_i^H \mathbf{r}_0 - \mathbf{u}_i^H (\hat{\mathbf{U}} \hat{\mathbf{B}} + \hat{\mathbf{s}}_{\mathbf{u}} \hat{\mathbf{f}}_{\mathbf{u}}^H) \hat{\mathbf{z}}_0 \\ &= \mathbf{u}_i^H \mathbf{r}_0 - \hat{\sigma}_i (\hat{\mathbf{z}}_0)_{(i)} \end{aligned}$$

The last relation follows from the orthonormality of the NPSVD. It follows that the updated residual is orthogonal to  $\mathbf{u}_i$  if we set  $(\hat{\mathbf{z}}_0)_{(i)} = \mathbf{u}_i^H \mathbf{r}_0 / \hat{\sigma}_i$ .

The Galerkin condition is justified in exact arithmetics where all search-vectors are orthogonal as described in sec. 7.4 p. 85. Thus, if the residual is orthogonal to all search-vectors, it must eventually be in the null-space of the operator.

In the presence of rounding errors, the reality is very different. If the conjugate gradient fails to converge on a few (say, in 10–30 iterations), the search vector has essentially lost any relationship with the starting vector. For fast convergence it is therefore highly desirable to use the largest NPSVD that will fit in memory.

The Galerkin orthogonalization removes a component of the residual of norm  $|\mathbf{u}_i^H \mathbf{r}_0|$ . This is substituted for a component in the search-vector of norm  $|\mathbf{u}_i^H \mathbf{r}_0| \cdot |(\hat{\mathbf{f}}_{\mathbf{u}})_{(i)}| / \hat{\sigma}_i$ . It follows that the residual norm can not grow if the triplets discarded during the restart satisfy  $|(\hat{\mathbf{f}}_{\mathbf{u}})_{(i)}| \leq \hat{\sigma}_i$ . From sec. 8.6 p. 100 this means that the singular vector has converged to a relative accuracy better than 1.

## 9.3 Unregularized solver

It is possible to combine the two restarting strategies in a single iterative solver. Since it is numerically advantageous to keep an NPSVD as large as possible, we can use implicit restarts as long as some singular triplets converges. In the unregularized case, it is always possible to limit the size of the NPSVD, since any triplet may be discarded at any time.

In practice, the difference between MINRES and Galerkin solutions are limited for triplets that converges to a high relative accuracy, i.e.  $|(\mathbf{f}_{\mathbf{u}})_{(i)}| \ll \hat{\sigma}_i$ . However, a large-scale problems that runs over many iterations may build up a number of triplets that converges very slowly. In this case, an explicit restart may give a significantly smaller residual norm than an implicit restart.

A simple solution is to monitor the norm resulting from using the two methods. If an explicit restart result in a residual norm that is significantly better than the Galerkin method, then such a step should be taken.

As an example, the default criteria used in BBTools is to use explicit restarting if the residual norm can be improved by a factor of 2. There is no rigorous argument for this, but it appears to work well in practice.

After a large number of iterations has been completed, the residual is stripped for components corresponding to large singular values. These triplets are easily identified from eq. (9.3b): the singular value  $\hat{\sigma}_i$  is small compared to  $\|\mathbf{A}\|_2$ .

Even if  $|(\mathbf{f}_{\mathbf{u}})_{(i)}|$  is large, these components should be removed from the residual. Certainly, this should happen if  $\hat{\sigma}_i \leq \tau$ , i.e. the accuracy of the matrix-vector product, but applications may choose to use a much larger threshold.

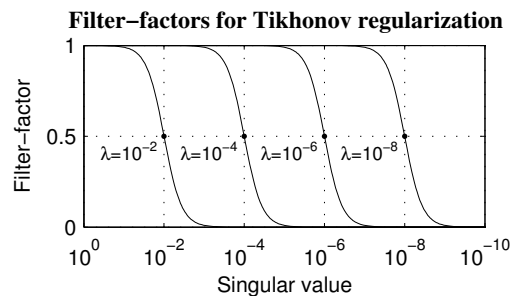
## 9.4 Regularized solver

At this point we are equipped to deal with the regularized solver. From sec. 5.5 p. 58 we would like a solution to the system  $\mathbf{A}\hat{\mathbf{x}} \approx \mathbf{b}$  written in terms of an SVD,  $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^H$  using a set of filter-factors:

$$(9.9) \quad \hat{\mathbf{x}} = \sum_{i:\sigma_i > 0} \mathbf{v}_i f_i \frac{\mathbf{u}_i^H \mathbf{b}}{\sigma_i}$$

In the current setting, we are mostly concerned with Tikhonov regularization as shown in fig. 9.1. In general, the filter-factors are most convenient to deal with when the singular values are given on a logarithmic scale.

**Figure 9.1:**  
Tikhonov filter-factors for various degrees of regularization.



There are two important things to note at this point:

- The regularized solution is linear in the filter-factors.
- The common filter-factors (see tab. 5.1 p. 59) are smooth if the singular values are plotted on a logarithmic scale.

This implies that the filter-factors can be interpolated linearly by sampling the filter-factors sparsely on a logarithmic scale. In other words, we may keep a few representative vectors instead of  $\hat{\mathbf{V}}$  and choose filter-factors (e.g. the regularization constant  $\lambda$ ) *after* the linear system has been solved.

This split the solving process into two separate steps:

- **Pre-solve:** solve the system using Galerkin orthogonalization, except only triplets that has converged to an acceptable tolerance is used. Also, instead of updating the solution, update the set of representative solutions.
- **Post-solve:** compute filter-factors, scale the representative solutions accordingly, and add.

The post-solving procedure is reduces to a matrix-vector product. On a modern computer this is, for any practical purposes, instantaneous. This allows the solver to be relatively expensive but still offers the user to adjust the regularization parameter interactively.

Generally, the singular triplets do not need to be very accurate, provided the filter-factors do not change too rapidly as a function of the singular value. Theorem C14 p. 152 guarantees that the numerical triplet is a sum of exact triplets and any components far from the approximated singular value must be small.

For Tikhonov regularization it is usually safe to orthogonalize and discard triplets that satisfy  $|(\mathbf{f}_{\mathbf{u}})_{(i)}| < c\hat{\sigma}_i$  where  $c \approx 0.1$ . However, the required accuracy is application dependent and should be tested empirically.

It should be noted that if  $c$  is very small<sup>1</sup>, the solver is forced to converge all singular triplets. In this case it may be more efficient, and memory conserving, to compute the SVD using the algorithm described in chapter 8.

## 9.5 Hybrid solver

Unfortunately, the requirement for relative accuracy of the singular triplets may prevent convergence. Generally, using a MINRES step is not an option, since the singular triplets are inaccurate.

However, after a large number of iterations, the residual mainly contains contributions from noise and small singular triplets that can not be resolved. This property hinges on the *Picard condition* [30]:

**Definition 9.1: (Discrete Picard condition)** *Let  $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^H$  be an SVD of an operator, and let  $\mathbf{Ax} = \boldsymbol{\lambda}$  be the expected measurement (i.e. the result measured in absence of noise). Then the discrete Picard condition is satisfied if the elements  $|\mathbf{U}^H\boldsymbol{\lambda}|_{(i)}$  decays faster than  $\sigma_i$  for sufficiently large  $i$ .*

Roughly speaking, the discrete Picard condition is satisfied if  $\mathbf{x}$  has limited bandwidth, as argued in sec. 5.2 p. 55.

Furthermore, the unconverged singular triplets tends to correspond to small singular values. This occurs because any component corresponding to large singular

<sup>1</sup>This is, for example, required for a truncated SVD.

values are picked up in a few iterations and is decoupled from the remaining component.

The last property implies that if a numerical triplet is expanded into components of exact singular triplets, then the singular values of the exact triplets tends to be much smaller than the numerical singular value of the triplet.

Therefore it is not dangerous to end the procedure with a MINRES step, and apply filter-factors according to the estimated singular values. This result in a *hybrid method* [30] where the exact algorithm used influence the result.

In practice, the number of steps required before the difference between the hybrid and true Tikhonov solution becomes negligible is problem dependent. Generally, the scheme works best if the norm of the measurement-noise is on the order of a few percent or more of the noiseless measurement, as is the case for most tomographic problems.

## 9.6 References

The literature on iterative solvers is vast because of the large number of practical applications. A pedagogical text-book that describes Krylov methods is [73], while a more complete references that includes algorithmic details is [4]. A good review with historical notes is [61].

Another text-book is [15], which also includes several Krylov methods and several other methods and references. It also raises the noteworthy point that Krylov methods are no panacea, and methods such as multi-grid may be more suitable for some applications.

One of the most successful general solvers is LSQR developed by Paige and Saunders [55]. The paper also includes an excellent discussion of bidiagonalization.

Hybrid methods are described in [30], which also include many references. The current knowledge is centered around methods where some results of early stopping is known.

The idea that, say, 1000 iterations is small compared to the size of the operator is not an ordinary theme in the literature. Hybrid methods are, at the moment, associated with some amount of handwaving. The author is unaware of any hybrid method that guarantees a solution that can be expressed independently of the algorithm in the presence of rounding errors.

*BBTools is a toolbox for MATLAB<sup>®</sup> implementing the algorithms described in the previous chapters and much more. It represents a major part of the work produced during the PhD.*

*This chapter is intended to be self-contained, and may be regarded as an introduction to BBTools.*

## 10.1 Introduction to BBTools

BBTools, or black-box toolbox, is a collection of routines for MATLAB<sup>®</sup> that implements almost everything discussed in the thesis in a way that should be accessible for most people familiar with MATLAB<sup>®</sup>.

The main focus of BBTools is to handle inverse ill-posed problems such as tomographic problems. However, the routines are general and can be used for a number of purposes, including:

- Inverse problems, regularized solutions of ill-posed problems and SVD analysis.
- Data mining, dimensionality reduction of large data sets.
- Iterative methods, research in algorithms based on Krylov subspaces.

The main reason the toolbox exists is that it can cope with problems that are much larger than any software package available to the author. It has been designed to utilize the available memory as much as possible.

### The meaning of a black-box

In the context of BBTools, a “black-box operator” (or simply “black-box”) is a set of functions that implements the matrix-vector products  $\mathbf{Ax}$  and  $\mathbf{A}^H\mathbf{x}$ . Here  $\mathbf{A}$  is the operator and  $\mathbf{x}$  is a normal MATLAB<sup>®</sup> vector (or matrix).

It can be useful to think of a black-box as an extension of sparse matrices. However, instead of being a “matrix with a lot of zero elements”, a black-box is “any matrix where matrix-vector products can be computed efficiently”. This class of operators is much broader than sparse matrices.

The operator is a “black-box” because the algorithms used to compute the products are known only to the implementation. The internal representation of the operator may have little resemblance with the elements of the matrix with the same linear transform.

**Example 10.1:** *The function `bbconvn` creates an operator whose internal representation consists of a Fourier transform of the convolution kernel. Fortunately, neither the user nor BBTools need to know this fact. (Except, of course, for `bbconvn`, which is part of BBTools).*

## How BBTools helps

In BBTools, the set of functions is encapsulated in an object that can be treated like a MATLAB<sup>®</sup>-matrix in many cases. In most other packages (including functions shipped with MATLAB<sup>®</sup> such as `lsqr`), a black-box operator must be given as a function (or set of functions).

Since a black-box can be manipulated like an ordinary MATLAB<sup>®</sup>-matrix, it is usually much easier to create and/or change an operator. Instead of reinventing the wheel, things such as Kronecker products and concatenations are available using the normal MATLAB<sup>®</sup> syntax.

Furthermore, BBTools provides a set of standard black-boxes to use as a starting point. This includes convolutions, Hadamard matrices, and basic operators such as zeros, diagonal matrices, and identity operators. Anyone familiar with MATLAB<sup>®</sup> knows how even simple operators can simplify a complex task.

The more general question “where do black-box operators come from?” is really two different questions:

1. Where do nature create problems with exploitable structure?
2. Where do the black-box operators used in BBTools come from?

We answer the first question with a quotation from [73]:

Large matrices ... usually arise indirectly in the discretization of differential or integral equations. One might say, that if  $m$  is large, it is probably an approximation to  $\infty$ . It follows that most large matrices of computational interest are simpler than their vast number of individual entries might suggest.

The second question is addressed in sec. 10.2.

## How black-box operators can be utilized

Black-box operators are only useful if they can be used to solve a problem. Although a matrix-vector product may be a useful operation in itself, there are many important iterative algorithms that are based solely on this operation.

From a practical point of view, the most useful procedure involves so called Krylov subspaces. BBTools is focused on the following problems:

- Solving large-scale SVD-problems.
- Computing regularized solutions to large-scale linear problems.

Theoretically, any operation performed on a matrix could be performed on a black-box operator. This is possible because the matrix corresponding to the linear transform can be computed using the matrix-vector product. From a pragmatic point of view, this would defeat the purpose of using black-box operators.

## 10.2 Central features

The fundamental unit in BBTools is the black-box operator. From this starting-point, BBTools targets 4 key areas in linear algebra.

## Creating and manipulating operators

The fundamental idea in BBTools is to encapsulate linear operations in operators. BBTools sports a number of features for creating and working with such operators:

- **Library of operators:** Many common operators are included to provide a starting point. Although some of these may occasionally be sufficient to model real-world problems, most operators are meant to provide “building-blocks, not solutions”. Most functions available in BBTools is provided in appendix D.
- **Operators can be combined:** Creating operators by combining simpler ones is standard in MATLAB<sup>®</sup>, but BBTools supports huge operators by storing an evaluation tree rather than explicit elements. This is transparent to the user.
- **Support for custom operators:** Operators can be highly specialized and require large programs to implement efficiently. If such programs are embedded in a black-box, they can be manipulated which often leads to better modularization and reuse of code.
- **Support for testing operators:** When complex operators are implemented, it is useful to have a set of standard functions for testing the correctness of the algorithms. This is a natural part of BBTools.

## SVD Analysis

The singular value decomposition (SVD) is one of the most important tools for analyzing linear systems. It forms a cornerstone of BBTools, due to its strong connection with regularized solutions of linear systems.

In its simplest form, the function `bbsvds` can replace `svds` in the common case where some of the triplets corresponding to the largest singular values are desired. It is usually faster, partly because `svds` does not utilize ARPACK in the most efficient way, and partly because BBTools is designed to exploit modern hardware more efficiently.

A bit more under the hood, we find some more drastic differences:

1. `bbsvds` computes singular values with an amount of memory that scales as  $\mathcal{O}(m + n + \min(m, n)k)$ , where  $k$  is the number of requested singular values. Although many packages can do this, BBTools do so without compromising the accuracy.
2. `bbsvds` can often compute more singular values/triplets than will fit in memory. This is accomplished without swapping data to and from disk; BBTools does not contain out-of-core routines. Instead, it works by trading memory for computational work. Rounding-errors may prevent BBTools from computing a full SVD of a large operator, but the extension can be very useful.

Computing singular values is easier than full singular triplets. If singular vectors are required, BBTools will normally use memory that scales as  $\mathcal{O}((m+n)k)$ . The last point remains true: it is still possible to compute more triplets than memory can hold.

### Solving ill-posed problems

The goal of a large linear problem is often to solve the system on the form  $\mathbf{Ax} = \mathbf{b}$ . Unfortunately, the system-matrix  $\mathbf{A}$  is ill-conditioned in many practical problems and  $\mathbf{b}$  have measurement-, rounding-, and representation-errors.

Regularization deals with these problems. BBTools is concerned with solutions that can be expressed in terms of an SVD of  $\mathbf{A}$  and a set of filter factors. This includes many standard techniques.

Unfortunately, the SVD of  $\mathbf{A}$  is infeasible to compute in practice, and we must resort to approximate methods. BBTools accomplishes the task in two steps:

1. `bbpresolve` An iterative solver, except the result is not returned as a vector (`bbsolve` can do that). Rather, the solution is split into a number of solutions, each corresponding to a set of local filter-factors.
2. `bbregsolve` Form a regularized solution by approximating the wanted filter-factors as a sum of the local filter-factors. The solution,  $\mathbf{x}$ , is then a weighted sum of the solutions computed by `bbpresolve`.

It is typically necessary to compute a number of regularized solutions in order to balance noise and accuracy. The L-curve method (`bbllcurve`), for instance, requires a number of such evaluations. BBTools supports this efficiently: `bbregsolve` costs almost nothing when it is fed with the output from `bbpresolve`.

The solver in BBTools is primarily intended for ill-posed problems. When a “real” (i.e. unregularized) solution is meaningful, it may be better to use other routines.

Note that preconditioning destroys the singular values of the operator, and therefore regularization. As a consequence, the code was developed to cope with more iterations than other solvers normally expect is necessary.

### Iterative routines

Black-box operators generally require iterative routines, and both the SVD routines and solvers described above are iterative. However, some problems are already handled well by existing MATLAB<sup>®</sup> routines. For instance, a function like `eigs` is excellent.

There are several cases where one may want to use the iterative solvers shipped with MATLAB<sup>®</sup>:

1. An accurate, unregularized, solution is required.
2. A test-battery of algorithms is needed for comparison or benchmarking.
3. A known algorithm must be used to allow reproducible results.

BBTools supports most iterative routines shipped with MATLAB<sup>®</sup>.



## Research in iterative routines

Since the 1960s we had libraries of canned routines for most dense problems in linear algebra [17]. The libraries have been rewritten to match the technology; we went from LINPACK and EISPACK to LAPACK [2], and later ScaLAPACK [10] emerged.

As computers get faster, we expect to handle larger problems, and this eventually leads to more use of iterative routines [73]. The literature is extensive, and many iterative solvers have been proposed; a good introduction can be found in [61].

Such a plethora exists because there is no single algorithm that consistently beats the competition. No canned routine have emerged where we can leave its implementation to specialist and invoke with  $A \setminus b$ , trusting that it “just works”.

BBTools should be helpful with research of such algorithms. For instance, it allows one to evaluate the effect of degraded accuracy and count products without changing the code of the solvers.

However, the most important aspect of BBTools is perhaps that it allows the end-user a familiar entrance to the iterative routines. The popularity of MATLAB<sup>®</sup> can be seen as an example of the importance; most of the functionality in MATLAB<sup>®</sup> is available as free libraries on the Internet. Nevertheless, MATLAB<sup>®</sup> is the de facto language for research in numerical linear algebra.

## 10.3 Working with black-box operators

BBTools is rooted in the idea that any linear transformation can be represented as an operator that acts on a column-vector<sup>1</sup>. While this may sound innocent, it implies a mental model that may be unfamiliar to many.

In most cases we do not think much about the meaning of a “matrix”. To work efficiently with BBTools, it is advantageous to distinguish between operators (linear transforms) and hypercubes (data arrays). The most immediate consequence of the policy is that any matrix/array must be converted to a column-vector before an operation can be applied.

**Example 10.2:** *Assume that IMG is an  $100 \times 100$  image. An operation might traditionally do something like the following:*

```
IMG2=5*IMG(1:50,20:90);
IMG2=convn(IMG2,K);
```

*Using BBTools this would instead look something like this:*

```
[A1,dim1]=bbredim([100,100],[1,20;50,90]);
[A2,dim2]=bbconvn(dim1,K);
A=A2*5*A1;
IMG2=reshape(A*IMG(:),dim2);
```

This may look unnecessarily complex, but it has the advantage that the steps are captured in A. This allows us to pass it as an argument to a function, which can be blissfully ignorant that IMG represents an image.

<sup>1</sup>One may also use row-vectors provided it is used consistently. This is often preferred by statisticians [52].

**Example 10.3:** *Continuing the last example, apply the adjoint operator on IMG2:*

```
IMG3=reshape(A'*IMG2(:),dim1)
```

While the adjoint could be worked out in a simple example like this, BBTools made it trivial because the adjoints of the two sub-functions has been worked out once and for all.

## Matrices as operators

The word “operator” is usually defined (in the current context) in a way similar to the following [25]:

... any symbol that indicates an operation to be performed.

That is, an operator is not a passive table of elements; rather it is something active like a function. This view is sometimes helpful in understanding BBTools.

Let us go back to basics: linear operators are functions that map a vector-space to another linearly. That is,  $T$  is a linear operator if it satisfies the following for all  $\tilde{\mathbf{x}}, \tilde{\mathbf{y}}$  in the input domain and scalars  $c$ :

$$(10.1) \quad T(\tilde{\mathbf{x}} + \tilde{\mathbf{y}}) = T(\tilde{\mathbf{x}}) + T(\tilde{\mathbf{y}})$$

$$(10.2) \quad T(\tilde{\mathbf{x}}c) = cT(\tilde{\mathbf{x}})$$

When the input and output domains of a linear operator have finite dimensions, then any element, say  $\tilde{\mathbf{x}}$ , in the vector-space can be written in terms of a basis:

$$(10.3) \quad \tilde{\mathbf{x}} = x_1\tilde{\mathbf{b}}_1 + x_2\tilde{\mathbf{b}}_2 + \dots$$

As long as the basis is given, we may refer to  $\tilde{\mathbf{x}}$  by giving the coefficients (or coordinates) of this basis. These are conveniently represented by a vector of coordinates:  $\mathbf{x}$ .

Any linear operation that maps a column-vector to another may be described uniquely in the form of a matrix. The transformation  $\tilde{\mathbf{y}} = T(\tilde{\mathbf{x}})$  may be computed as a matrix-vector product  $\mathbf{y} = \mathbf{A}\mathbf{x}$ .

It is a good idea to keep in mind that an object is represented by a vector in terms of a basis. In many cases the vector is simply related to the basis, and it is easy to forget it exists. If an object is an image, for example, each element may refer to a pixel.

## Matrices as containers

Matrices are used for other things than describing linear operators. A matrix may hold any kind of data which can be stored in a 2-dimensional table. Common examples are images and pairwise relations such as correlation coefficients, forces, distances, etc.

BBTools is primarily concerned with data when operators are acting on them. Consider an image where some common editing is wanted, say cropping or sharpening. Although BBTools can create operators that performs such operations, it does not understand the notion of an image.

To apply the operation the data must be in canonical form, i.e. a vector. That is, a data-matrix it must be expanded in terms of a basis before BBTools can do anything with it. We shall use the term hypercube to emphasize that data-matrices and other arrays are the same to BBTools. We could have used a number of different words:

1. Hypercubes
2. Multi-dimensional arrays
3. Data matrices

To confuse matters further, a matrix may also represent a collection of column vectors, e.g.  $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots]$ . In this case an operator may be applied en masse:  $\mathbf{AX} = [\mathbf{Ax}_1, \mathbf{Ax}_2, \dots]$ .

It may occasionally be difficult to determine what category a matrix belongs to. From a practical point of view it usually suffice to distinguish between operators and data-arrays.

### Advantages and disadvantages

BBTools assumes that linear transformations are applied by creating an operator and applying it to the data, which has the form of a vector. There are several advantages with this approach:

- The implementation of an operator is separated from its invocation. This allows one to use standard routines and concentrate on the modelling problem (i.e. creating the operator) rather than the algorithms required for their solution. item Hypercubes are treated as any kind of data. BBTools does not care if a vector represents a rectangular image, a round one, or contains holes.
- Algorithms can be closer to their “text-book” versions than what is normally seen in practical code. This speeds up development, makes it relatively easy to analyze an algorithm, and facilitates debugging. The algorithms shown in this thesis are examples of this concept.
- Vectors are stored in matrices that can be manipulated via high-performance BLAS (see sec. 8.2 p. 96).
- The creation of “the adjoint operation” (i.e. computing  $\mathbf{A}^H \mathbf{x}$ ) only needs to be considered at the level of fundamental building-blocks. Operators build from these automatically supports this operation.
- BBTools keeps track of the accuracy of the matrix-vector product,  $\tau$ . Again, this only needs to be determined at the level of building-blocks. If necessary, the accuracy can be estimated from the operator directly.

The downside is that it is necessary to handle data manually. One of the more annoying examples is the necessity to keeping track of the dimensions of a hypercube. Functions that create operators corresponding to actions on a hypercube (e.g. `bbconvn`) often require the dimension as an input argument and often return the output dimension as a result.

## 10.4 Compatibility

One of the major design-goals was to be as compatible with MATLAB® as possible. Once a black-box has been created, it should act as closely as possible as an ordinary matrix. BBTools was designed to blend in with ordinary MATLAB® programs as unobtrusively as possible.

The main computations that black-box operators lacks is operations that works on individual elements. This implies that functions such as `abs` and `A.^c` are not supported.

### Operator combinations

Many common manipulations of MATLAB®-matrices are supported by BBTools, as shown in tab. 10.1. While the list may appear short, it covers the majority of the operations used in practical programs.

Syntax	Function	Description
<code>A+B</code>	<code>plus(A,B)</code>	Addition
<code>[A,B]</code>	<code>horzcat(A,B)</code>	Horizontal concatenation
<code>[A;B]</code>	<code>vertcat(A,B)</code>	Vertical concatenation
<code>A/c</code>	<code>mrdivide(A,c)</code>	Division by scalar
<code>A.'</code>	<code>transpose(A)</code>	Transposition
<code>A'</code>	<code>ctranspose(A)</code>	Complex conjugate transpose
<code>+A</code>	<code>uplus(A)</code>	Unary plus
<code>-A</code>	<code>uminus(A)</code>	Unary negation
<code>A*B</code>	<code>mtimes(A,B)</code>	Matrix multiplication (see below)
	<code>feval(A,B)</code>	Evaluate $A*B$ (discouraged)
	<code>feval(A,B,'transp')</code>	Evaluate $A'*B$ (discouraged)
	<code>conj(A)</code>	Complex conjugate
	<code>real(A)</code>	Complex real part
	<code>imag(A)</code>	Complex imaginary part
	<code>kron(A,B)</code>	Kronecker tensor product
	<code>repmat(A,...)</code>	Replicate and tile an operator

**Table 10.1:**  
Basic operations compatible with black-box operators.

### Matrix multiplication

Matrix multiplication is supported via the normal syntax `A*B` and the corresponding function `mtimes(A,B)`. However, to maximize compatibility there are special rules that applies to this product:

1. If either `A` or `B` is a MATLAB®-matrix, the product will be evaluated and returned as a normal MATLAB®-matrix.
2. If either `A` or `B` is a scalar, which can be interpreted as a  $1 \times 1$ -matrix, then the product will also be evaluated.
3. In any other case it returns a black-box operator corresponding to the product `A*B`.

The convention allows a black-box operator to replace a MATLAB®-matrix in many circumstances. If `A` is a black-box operator and `x` is a normal matrix, then `A*x`

evaluates the matrix-vector product, while `A*blackbox(x)` creates a black-box operator.

If a black-box is desired without evaluation, the function `bbprod` may be used instead of `mtimes`. This is guaranteed to return a black-box operator. Conversely, `bbmult` always return a MATLAB®-matrix. However, this is not possible if both inputs are black-box operators.

The forms using `feval` exist for compatibility purposes only. While this allows black-box operators to be used with some existing programs, the use should be discouraged.

### Complex operations

BBTools supports complex operators everywhere. However, if `A` is a complex operator, then `B=real(A)` or `B=imag(A)` creates an operator that evaluates two matrix-vector products with `A` to evaluate a matrix-vector product with `B`. If these functions are nested inside each other, this may lead to exponential explosion.

### Operator information

The ordinary functions used to get basic information about an operator are also supported, as shown in tab. 10.2. However, `disp`, `display` and, by extension, the result printed by a computation that evaluates to a black-box operator, is not the elements of the matrix but the evaluation tree.

**Table 10.2:**  
Basic information  
about a black-box  
operator.

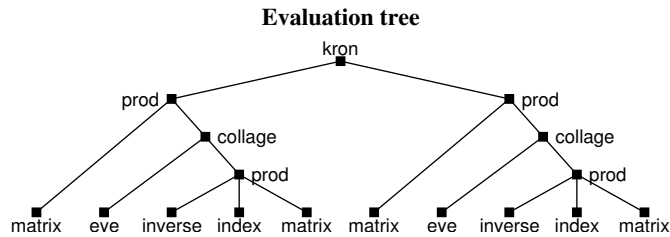
Function	Description
<code>size(A)</code>	Dimension of operator
<code>isreal(A)</code>	true if <code>A</code> is real
<code>isa(A, str)</code>	true if <code>A</code> is of type <code>str</code>
<code>disp(A)</code>	Display evaluation tree (as text)
<code>display(A)</code>	As <code>disp</code> , except the variable name is printed

**Example 10.4:** *One of the most complex functions in BBTools is `bbrescale`, which creates a black-box that rescales a hyper-cube to one with another dimension. For example, the command `A=bbrescale([100,100],[200,200],'spline')` creates an operator,  $\mathbf{A} \in \mathbb{R}^{(200 \cdot 200) \times (100 \cdot 100)} = \mathbb{R}^{40000 \times 10000}$ , that scales a  $100 \times 100$  image to  $200 \times 200$  using cubic spline interpolation. The command `disp(A)` prints the following evaluation tree:*

```
kron 40000x10000
> prod 200x100
  > matrix 200x200 (sparse)
  > collage 200x100
    > eye 100x100
    > prod 100x100
      > inverse 100x100 (LU substitution)
      > index 100x100
      > matrix 100x100 (sparse, central difference)
  > prod 200x100
    > matrix 200x200 (sparse)
    > collage 200x100
      > eye 100x100
      > prod 100x100
        > inverse 100x100 (LU substitution)
        > index 100x100
        > matrix 100x100 (sparse, central difference)
```

*A more traditional (and less informative) rendition is given in fig. 10.1.*

**Figure 10.1:**  
Evaluation-tree for  
an operator created  
by `bbrescale`.



## Norms

Analysis of linear systems often start with norms. While BBTools has powerful functions for SVD-analysis, the ordinary functions in MATLAB<sup>®</sup> also works with black-boxes as shown in tab. 10.3.

**Table 10.3:**  
Analysis support of  
black-box opera-  
tors.

Function	Description
<code>norm(A)</code>	Compute norm of an operator (2-norm only)
<code>normest(A)</code>	Rough estimate of 2-norm
<code>normest1(A)</code>	Good estimate of the 1-norm

The exception is the Frobenius norm. While this norm is often used because it is easy to compute for matrices (sparse or full), black-box implementations rely on iterative procedures to accomplish the same task. Unfortunately, it is difficult to estimate the Frobenius-norm reliably, and the author is unaware of good algorithms that can accomplish this task in the general case.

Note: BBTools includes the function `bbnormest`, which should be used instead of `normest`. It supports both 1-, 2-, and  $\infty$ -norms, and usually provides a much better accuracy for substantially less work than `normest`.

The function `norm` reverts to `bbnormest` with a warning for the 1- and  $\infty$ -norms. Although `bbnormest` (which is, in turn, based on `normest1`) is usually very accurate, there are rare cases where it may fail.

## Iterative routines

Some care have been taken to allow the iterative routines shipped with MATLAB<sup>®</sup> to work with black-box operators. MATLAB<sup>®</sup> usually assumes that these functions are intended for sparse matrices, but they work for black-box operators as well.

**Table 10.4:**  
Iterative routines  
supported by  
BBTools (param-  
eters are omitted).

Function	Description
<code>eigs</code>	Find a few eigenvalues and eigenvectors
<code>svds</code>	Find a few singular values and vectors
<code>lsqr</code>	Conjugate Gradients on the Normal Equations
<code>bicg</code>	BiConjugate Gradients
<code>bicgstab</code>	BiConjugate Gradients Stabilized
<code>cgs</code>	Conjugate Gradients Squared
<code>gmres</code>	Generalized Minimum Residual
<code>minres</code>	Minimum Residual
<code>pcg</code>	Preconditioned Conjugate Gradients
<code>qmr</code>	Quasi-Minimal Residual
<code>symmlq</code>	Symmetric LQ

## Miscellaneous

Finally, tab. 10.5 lists a few miscellaneous functions that do not fit in any of the previous categories.

**Table 10.5:**  
*Miscellaneous  
functions sup-  
ported by BBTools.*

Function	Description
<code>sum(A)</code>	Column sum (or rows)
<code>mean(A)</code>	Column average (or columns)

## 10.5 References

BBTools, and its manual, is freely available online [33]. Once installed, the documentation is available in the MATLAB<sup>®</sup> environment similar to the toolboxes available from Mathworks. A list of most functions provided by BBTools is given in appendix D p. 157.

The toolbox “Regularization Tools” [29], by Dr. Per Christian Hansen, is a fairly complete toolbox for MATLAB<sup>®</sup>, and contains numerous tools and test problems. It is very useful for people who want an introduction to the topic of regularization and ill-posed problems. The software and its manual are available online together with a technical report [31].

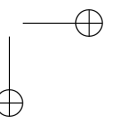
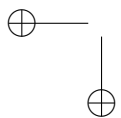
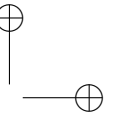
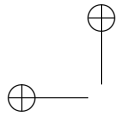
The toolbox is mainly concerned with techniques for small dense systems. There are, however, a few iterative routines and some functions which work independently of the solutions (e.g. computation of filter-factors).

The MATLAB<sup>®</sup>-toolbox *MORE* Tools offers a different approach to implement operators [38]. Instead of considering an operator as a black-box, it uses object oriented programming. That is, an operator is implemented as a class, and is not limited to matrix-vector products. In fact, the operations supported by *MORE* Tools increase with the capabilities a particular class implements.

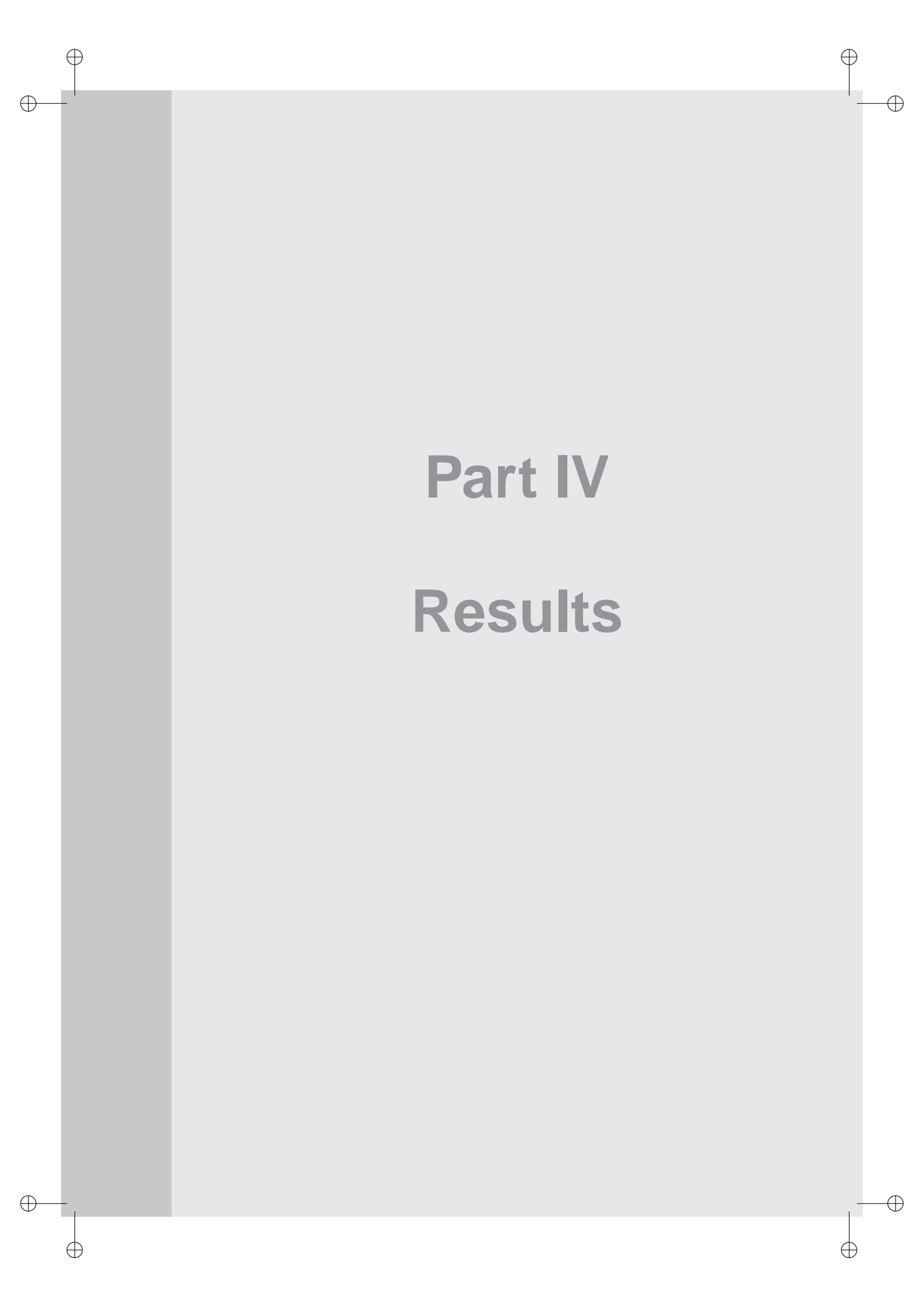
Both BBTools and *MORE* Tools can combine operators in an evaluation tree. However, the main emphasis of *MORE* Tools is abstraction and infrastructure, whereas BBTools is efficiency, minimalism, and problem solving.

For example, *MORE* Tools does not require an explicit conversion between hypercubes and vectors; it understands the difference once the class has been implemented. On the other hand, it is not necessary to teach BBTools there is a difference.

Someone who mainly uses BBTools to combine operators, or is looking for a framework for implementing general methods, may be served best by *MORE* Tools. On the other hand, BBTools may be preferable to someone who wants to solve a specific problem.

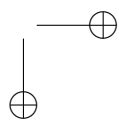
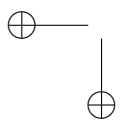
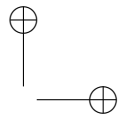
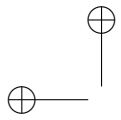






**Part IV**

**Results**



Finally, we present the results of using regularized reconstruction. We do so using the phantoms from chapter 3, and compare the regularized reconstruction with the traditional methods.

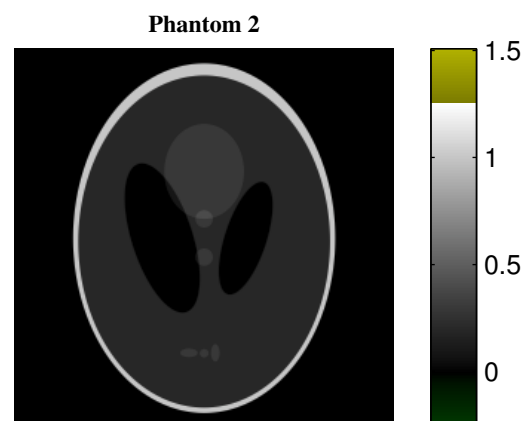
Since the goals described in chapter 5 do not lend themselves to quantitative analysis, the reconstructed images must stand on their own. Hopefully, the power of regularized reconstruction is evident from these images.

## 11.1 Introduction

One thing might as well be admitted from the start: the method chosen in chapter 5 was not chosen to optimize quantitative accuracy. This brings up to the disturbing fact that an analysis of the quantitative accuracy is pointless; much better methods for this purpose can be found in the literature.

Therefore, the only results given in this chapter will be images produced by the algorithms. While this may be disappointing for those who are mathematically oriented, it is good news for the medical users targeted in this project: there will not be any mathematics in the chapter.

The starting point is the Shepp-Logan phantom introduced in chapter 3. For convenience we reproduce it in fig. 11.1



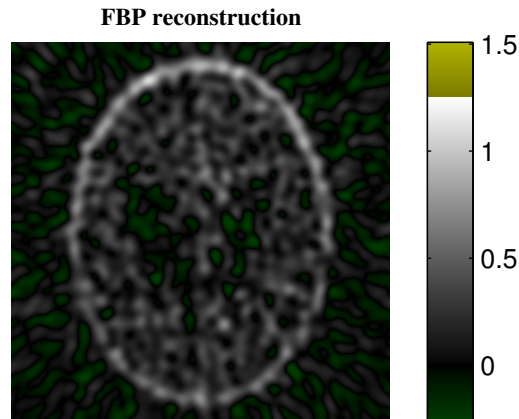
**Figure 11.1:**  
The classical  
Shepp-Logan  
phantom repro-  
duced from fig. 3.2.

## 11.2 Low-count reconstruction

Starting with the most difficult case, we return to the case where only 20,000 counts are acquired. This represents a typical scenario in SPECT, where it is typical to acquire about 500,000 counts in medical scanning. In 2D reconstruction, these counts are distributed on a number of slices, which is determined by the size of the acquisition matrix.

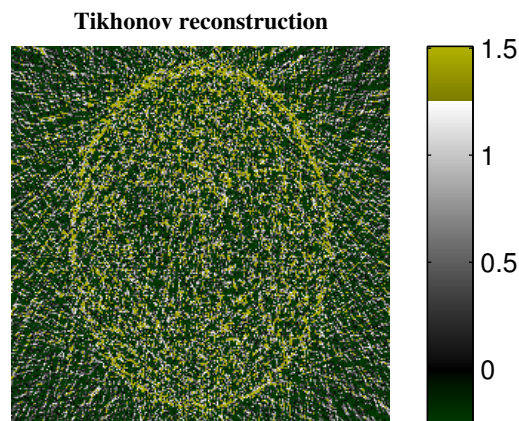
In the current scenario, the acquisition matrix typically consists of 64 slices. However, the central (and most interesting) slices will usually get more counts than the outer slices. A typical reconstruction of a slice using filtered back-projection is shown in fig. 11.2.

**Figure 11.2:**  
The sinogram in  
fig. 3.12 recon-  
structed using FBP.



Invoking BBTools, we may reconstruct the same image using Tikhonov regularization. Since we intend to apply a post-filter with a Gaussian kernel, the regularization parameter should deliberately be chosen to provide an under-regularized solution. For the current example<sup>1</sup>, the result may look like fig. 11.3.

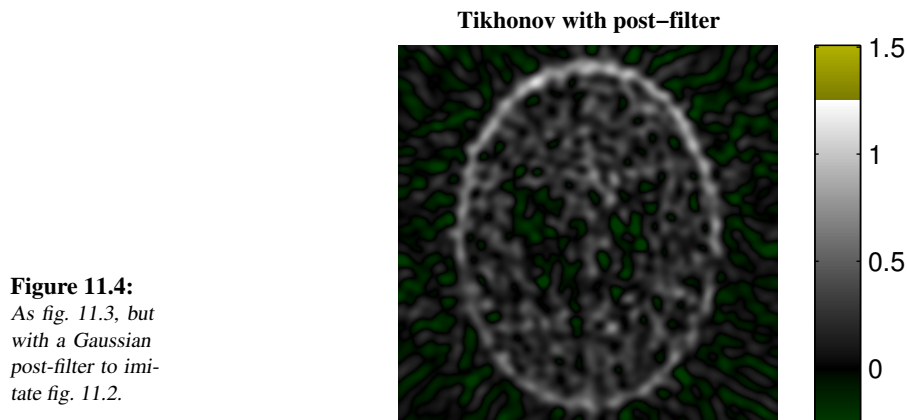
**Figure 11.3:**  
An under-  
regularized re-  
construction of  
fig. 11.2.



As described in sec. 9.4 p. 110 this is a two-step procedure: first a computationally expensive presolving procedure is performed, and then the regularization-constant can be chosen interactively.

After applying a post-filter, which was chosen to match the resolution of the FBP-reconstruction in fig. 11.2, the under-regularized image (fig. 11.3) is transformed into fig. 11.4. At this point it might be useful to refresh one of the goals from chapter 5: the reconstruction algorithm should be backwards compatible, i.e. it should be possible to reconstruct images that are similar to the traditional algorithm.

<sup>1</sup>The regularization-constant was chosen according to the L-curve criteria [30], which occurs around  $\lambda \approx 0.021\|\mathbf{A}\|_2$  in this case. However, we shall generally assume that the parameter is fixed or chosen interactively.



**Figure 11.4:**  
As fig. 11.3, but  
with a Gaussian  
post-filter to imi-  
tate fig. 11.2.

To judge whether this goal has been achieved, we merely need to compare whether fig. 11.2 and fig. 11.4 are similar. Those who agree they are similar also agree that the algorithm is backwards compatible as desired.

### Comparison of point-spread functions

As we also mentioned<sup>2</sup>, in chapter 5, there would be no progress if backwards compatibility was the sole goal; in that case we might as well use the old algorithm in the first place.

However, a closer look on the two algorithms do reveal a difference. Recall that a point-spread function (PSF) is the reconstructed image of a point in absence of noise. A number of point-spread functions for the FBP-reconstruction that produced fig. 11.2 are shown in fig. 11.5, while the same PSFs corresponding to fig. 11.4 are shown in fig. 11.6.

Comparing these we may draw two conclusions:

1. The PSFs of post-filtered Tikhonov-reconstruction are more homogeneous compared to FBP. This is most expressed near the edges.
2. The negative values are reduced.

Both are very desirable goals.

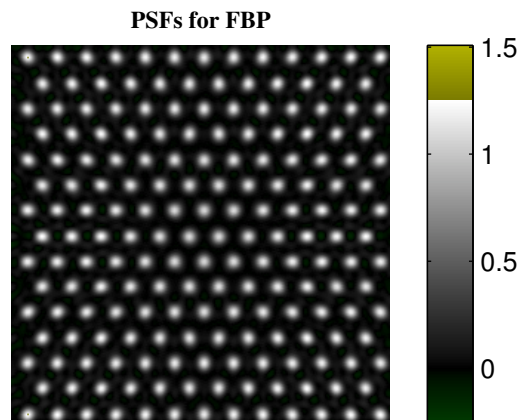
### Comparison with EM

The comparison would be incomplete without including an EM-reconstructed image. Since the EM-algorithm is non-linear, and the PSFs would therefore be meaningless, this comparison can only be performed on images.

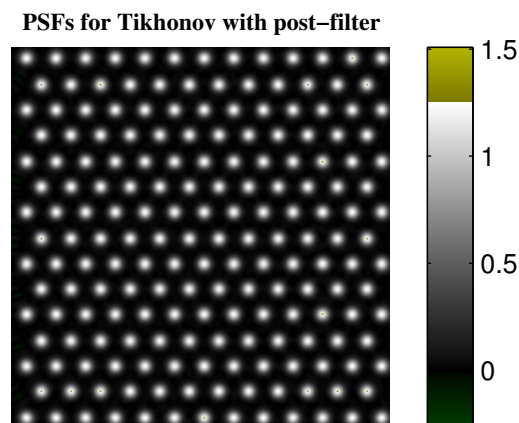
To make the comparison as fair as possible, the EM-image in fig. 11.7 was reconstructed using the same post-filter as the Tikhonov image in fig. 11.4. Although a sieved image might yield slightly different result, choosing a set of parameters would be a subjective choice.

<sup>2</sup>Section 5.4 to be precise.

**Figure 11.5:**  
Point-spread functions for the reconstruction in fig. 3.12.



**Figure 11.6:**  
As fig. 11.5, but using regularized reconstruction with post-filtering.



At first sight, the EM-image may appear “nicer” than the other images. However, it is questionable whether it actually contains more information. There is also a tendency to create false features that look natural.

Perhaps the one quality that makes the EM-image look better is that it does not contain negative counts. We have consistently shown the negative values also, but for the purpose of comparison, fig. 11.8 is clipped so negative values are black. It is now much more questionable whether the EM reconstruction, fig. 11.7, offers a significant advantage over FBP in fig. 11.8.

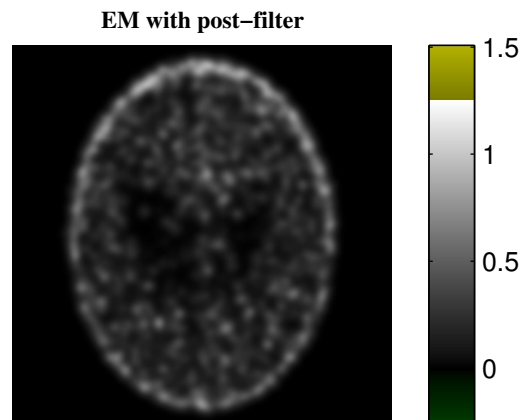
### Invoking Tikhonov

When we are assured that we can approximate the FBP-algorithm closely, it is interesting to see the effect of the Tikhonov regularization. At this point we have two options:

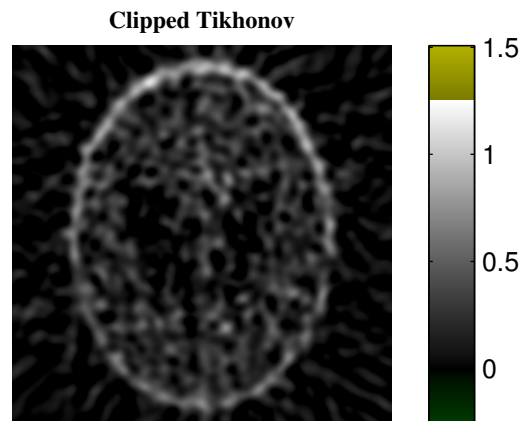
- Suppress more noise at the same resolution.
- Increase resolution at the same noise-level.

The image in fig. 11.9 shows an example of more noise-suppression. In this case, the regularization-constant was set 10 times higher than the previous image. The

**Figure 11.7:**  
As fig. 11.12, but reconstruction using the EM-algorithm with the same post-filter used in fig. 11.3.

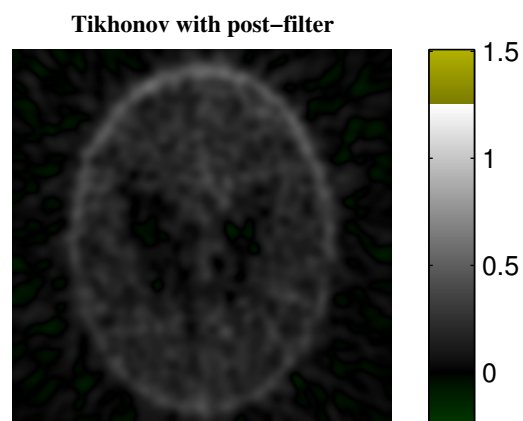


**Figure 11.8:**  
A reproduction of fig. 11.3, except negative values have been clipped to zero.



noise has been suppressed as desired, without any obvious loss of resolution.

**Figure 11.9:**  
As fig. 11.4, but with a regularization-constant that is 10 times higher.

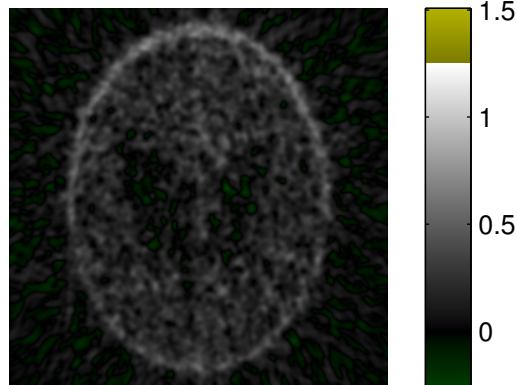


Tikhonov regularization suppress noise by making the images less extreme. While this avoids artifacts, it also tends to make real features hard to detect.

Another possibility is to increase the resolution once the noise has been suppressed. For specificity, we shall use a post-filter with twice the resolution, i.e.

the FWHM of the PSFs is halved. The result is shown in fig. 11.10, which shows that the resolution is improved as desired.

**Tikhonov with post-filter**



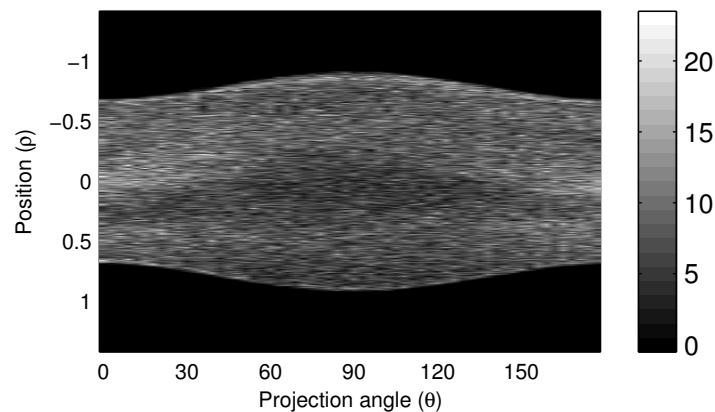
**Figure 11.10:**  
As fig. 11.9, except  
the resolution has  
doubled.

The reconstruction parameters on these examples were chosen without any rigorous criteria. While this is a weakness, there are only two parameters: the resolution and noise-suppression. Both may be selected while the image is updated interactively.

### 11.3 Medium-count reconstruction

We now turn to a scenario that lies in between a low-count and high-count scenario. While we use the same phantom, for easy comparison, we will work with a measurement where about 100,000 counts are acquired as shown in fig. 11.11.

**Measured sinogram (99435 counts)**



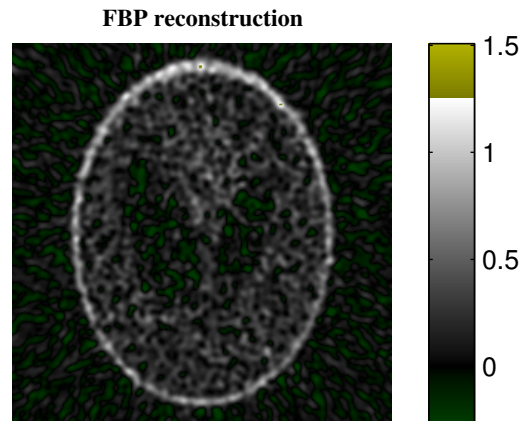
**Figure 11.11:**  
Sinogram of the  
Shepp-Logan  
phantom with  
approximately  
100,000 counts.

While this number of counts is difficult to achieve in clinical SPECT, it is very low for a PET-scanning. However, it is easier to see the effects of reconstruction when a limited number of counts are used, and it is a more difficult problem; if the low-count problem can be handled, then the high-count problem is easy.

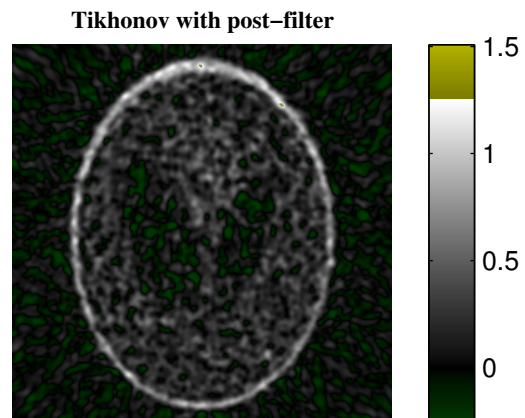
For comparison, fig. 11.12 shows a FBP-image, reconstructed to match the resolution in the Tikhonov-enhanced image in fig. 11.10. Again, this can be imitated by an appropriate filter (fig. 11.13) and the resolution can be increased by suppressing extremes (fig. 11.14).



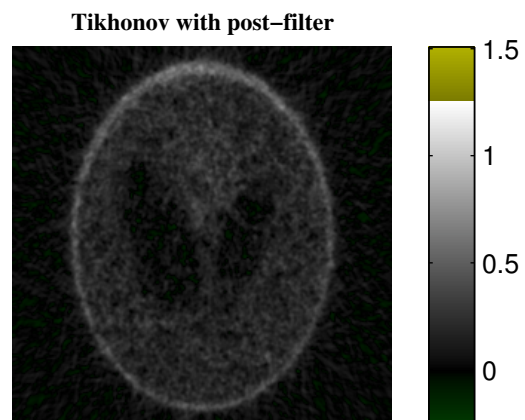
**Figure 11.12:**  
A FBP-  
reconstruction of  
the sinogram in  
fig. 11.11.



**Figure 11.13:**  
Post-filtered regu-  
larized solution to  
imitate fig. 11.12.



**Figure 11.14:**  
The same as  
fig. 11.13, ex-  
cept more filtering  
is performed us-  
ing regularization  
and less by post-  
filtering.

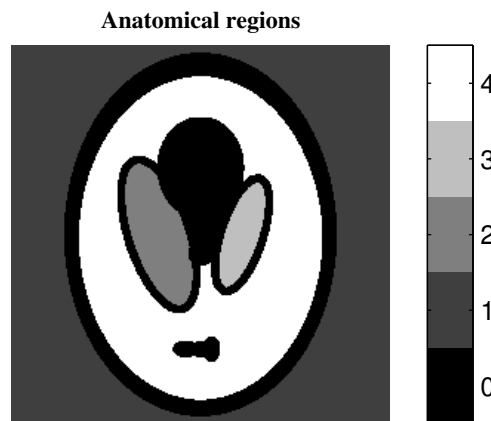


## 11.4 Using anatomy in reconstruction

So far we only mentioned anatomy in chapter 5, where it was stated that it would be easy to incorporate this into the reconstruction. It is time to make good on this promise.

First, we need to define a set of anatomical regions that are known to have constant concentrations. In a real workflow, this would typically be accomplished either by drawing regions on another set of images, e.g. MR, or using a template. In this example we cheat, and use regions based on the original phantom as shown in fig. 11.15.

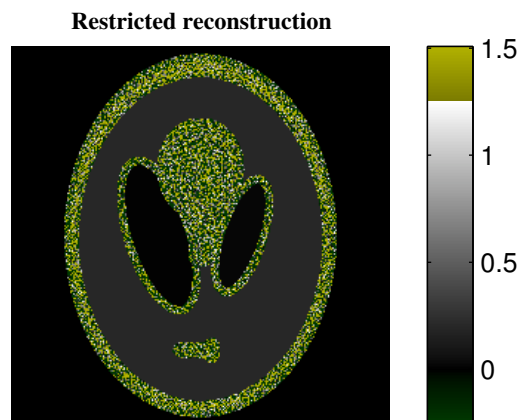
**Figure 11.15:**  
Anatomical regions as they might have been defined by a user. Region 0: nothing is assumed, Region 1: known to be zero, Region 2,3,4: constant.



The regions shown in fig. 11.15 are well within the region on the original phantom. That is, the regions are “safe” and need not be perfectly delineated. In this sense, the example is realistic.

As described in sec. 5.6 p. 61, we may now reconstruct the image under the constraints defined by the regions. An under-regularized solution is shown in fig. 11.16. Clearly, the constrained regions are large enough to be determined accurately.

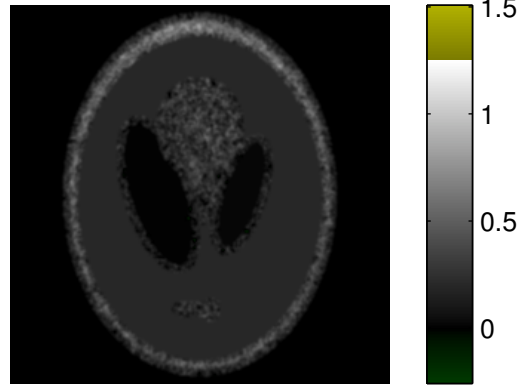
**Figure 11.16:**  
Underregularized solution to determine the concentration of the anatomical regions.



Finally, the known regions may be subtracted out while the remaining image is reconstructed using a more regularized filter. This leads to fig. 11.17, which is

the culmination of the project: a tomographic image reconstructed using anatomy and regularization.

**Final reconstruction**



**Figure 11.17:**  
*The final image, re-constructed using the anatomy and regularization.*

# Conclusion

## Conclusion

After a general introduction to the scanners used in emission tomography, we identified the main weakness of the current image formation methods: the resolution.

Currently, nearly all images in emission tomography are produced using either filtered back-projection (FBP) or expectation maximization (EM), both of which have been available for a long time. Much work has been done to improve these algorithms, but penetration into the medical fields has been limited.

Traditionally, the criteria used for success has been to reduce a residual 2-norm, energy, root-mean-square, or other equivalent measure. Instead, we identified a number of requirements posed by the intended users. This resulted in a list where the traditional criteria did not figure prominently.

Many, or even most, of these objectives are either unmeasurable or prohibitively expensive and longwinded to test. In order to make progress, the method to be developed was required to be backwards compatible. Since images obtained by a classical method can be imitated, this would ensure that the new method would be acceptable for the user community.

To this end, regularization via filter-factors was chosen as the method to be pursued. It was proved that methods in this class remained competitive with a much larger class of algorithms, even if only the residual is considered.

We continued by developing numerical algorithms to compute the solutions. Although regularization is textbook material, implementing a viable algorithm was challenging. Nevertheless, an algorithm was developed where regularized reconstructions of 2D images could be performed on consumer laptops.

Previous algorithms for computing regularized solutions were unable to cope with problems on the scale required for practical medical reconstruction. This may be surprising, since the technique is widely used in many diverse fields. Conversely, the new algorithms may be useful in applications reaching beyond medicine.

The regularized reconstruction algorithm was shown to produce images without creating random artifacts, i.e. brain-looking features, other than those created by the traditional method. This is important, since it allows the resolution to be enhanced at the cost of suppressing elevated or lowered regions.

To reproduce the traditional algorithm, a post-filter was applied to an under-regularized solution. The result is visually indistinguishable from the traditional algorithm it imitates. Furthermore, the reconstruction algorithms allow the user to balance the amount of filtering (either regularization or post-filtering) in real-time, once the initial processing is complete. Therefore, the user can start with the classical image, and choose a different setting by inspection.

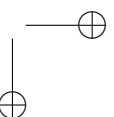
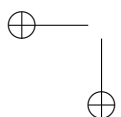
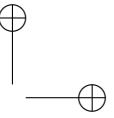
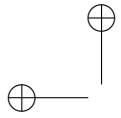
The new software made it possible to incorporate anatomical information by constraining certain regions to constant concentrations. The practical consequence of this is that no assumptions were required in regions where the user would be unsure of the result, e.g. near lesions.

## Scientific contributions

There are several novelties in the project:

1. Rigorous bounds were established on linear inversion of Poisson-distributed measurements.
2. Algorithms were developed to handle large-scale SVD computations and computing regularized solutions for large-scale linear systems using arbitrary filter-factors.
3. Software was produced to make the algorithms accessible in a familiar MATLAB<sup>®</sup> environment.
4. Regularized reconstruction was demonstrated to be viable on consumer hardware.
5. Finally, we showed that anatomical constraints could be incorporated in the framework.

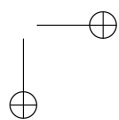
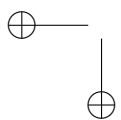
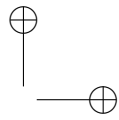
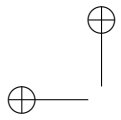
It should be noted that most of these results have applications outside tomographic reconstruction. Indeed, regularization is fundamental for a large number of inverse ill-posed problems in e.g. geology and helioseismology. An extension to the range of problems that could be handled by previously available software is interesting for all such applications.





**Part V**

**Appendix**





## Addendum to WLS

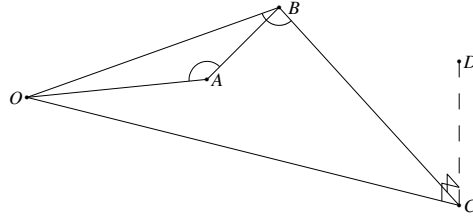
This appendix contains technical proofs used in the analysis of weighted least-squares in chapter 6.

## A1 Image-space bound

**Lemma A1:** **(Point-distances on convex surfaces)** Let  $O$  be the origin of a Cartesian coordinate system,  $B \in \mathcal{S}_1$  the point on a convex surface  $\mathcal{S}_1$  closest to  $O$ , and  $A \in \mathcal{S}_2$  the point closest to  $O$  on another convex surface  $\mathcal{S}_2$ .

Assume neither surfaces contains  $O$  in their interior. Finally, let  $D$  be a point on  $\mathcal{S}_1$  and  $C$  the orthogonal projection of  $D$  into the plane  $OAB$  as shown in fig. A.1. Then we have the inequality  $|BD|^2 + |AB|^2 \leq |OD|^2 - |OA|^2$ .

WLS-solutions in image-space



**Figure A.1:**  
Geometry of the points in the distance lemma.

PROOF

Since  $A$  and  $B$  are part of the convex surface  $\mathcal{S}_2$ , where  $A$  by definition is closest to  $O$ , it follows that  $\angle OAB \geq 90^\circ$ . By the same reasoning on  $\mathcal{S}_1$ , we see that  $\angle OBC \geq 90^\circ$ .

From this we have  $|OC|^2 \geq |OB|^2 + |BC|^2$  and  $|OB|^2 \geq |OA|^2 + |AB|^2$ , which combines into  $|AB|^2 + |BC|^2 \leq |OC|^2 - |OA|^2$ . Since  $CD$  is orthogonal to the  $OAB$ -plane we further have  $|BD|^2 = |BC|^2 + |CD|^2$  and  $|OD|^2 = |OC|^2 + |CD|^2$ . This finally gives the inequality  $|BD|^2 + |AB|^2 \leq |OD|^2 - |OA|^2$ . ■

**Theorem A2:** Let  $\ell_P$  and  $\ell_{WLS}$  be as defined in sec. 6.3 p. 67. Furthermore, let the penalty-function be given by  $r(\hat{\mathbf{x}}) = \|\mathbf{L}\hat{\mathbf{x}}\|_2^2$ .

Let  $\hat{\mathbf{x}}$  be the reconstructed image, given by def. 6.1 p. 68, using the Poisson-likelihood. Further assume there are regularization-constants,  $\alpha_a$  and  $\alpha_b$ , resulting in WLS-solutions,  $\mathbf{x}_a$  and  $\mathbf{x}_b$ , which satisfies:

$$\begin{aligned} \ell_P(\mathbf{A}\mathbf{x}_a, \mathbf{b}) &\leq \ell_P(\mathbf{A}\hat{\mathbf{x}}, \mathbf{b}) \leq \ell_P(\mathbf{A}\mathbf{x}_b, \mathbf{b}) \\ \ell_{WLS}(\mathbf{0}, \mathbf{b}) &< \ell_{WLS}(\mathbf{A}\hat{\mathbf{x}}, \mathbf{b}) \leq \ell_{WLS}(\mathbf{A}\mathbf{x}_b, \mathbf{b}) \end{aligned}$$

If the reconstruction algorithm is continuous in the regularization parameter,  $\alpha$ , then there exist WLS-solutions,  $\mathbf{x}_1$  and  $\mathbf{x}_2$ , that satisfy:

$$\begin{aligned} r(\hat{\mathbf{x}} - \mathbf{x}_1) + r(\hat{\mathbf{x}} - \mathbf{x}_2) &\leq r(\mathbf{x}_2) - r(\mathbf{x}_1) \\ r(\mathbf{x}_1) &\leq r(\hat{\mathbf{x}}) \leq r(\mathbf{x}_2) \end{aligned}$$

PROOF

Since the reconstructed image is a continuous function of  $\alpha$ , there exists a regularization parameter,  $\alpha_a \leq \alpha_2 \leq \alpha_b$  resulting in a WLS-image,  $\mathbf{x}_2$ , on the Poisson-surface:  $\ell_P(\mathbf{A}\mathbf{x}_2, \mathbf{b}) = \ell_P(\mathbf{A}\hat{\mathbf{x}}, \mathbf{b})$ . Since  $\hat{\mathbf{x}}$  by definition minimizes the penalty-function, we have the relation  $r(\hat{\mathbf{x}}) \leq r(\mathbf{x}_2)$ .

Similarly, there is a WLS-solution,  $\mathbf{x}_1$ , on a WLS-surface containing the Poisson-solution:  $\ell_{\text{WLS}}(\mathbf{A}\mathbf{x}_2, \mathbf{b}) = \ell_{\text{WLS}}(\mathbf{A}\hat{\mathbf{x}}, \mathbf{b})$ . As before we have  $r(\mathbf{x}_1) \leq r(\hat{\mathbf{x}})$  by definition.

Now let  $r(\mathbf{x}) = \|\mathbf{x}\|_2^2$ , and apply lemma A1 with  $\mathbf{x}_2$  as point  $D$ ,  $\hat{\mathbf{x}}$  as point  $B$  and  $\mathbf{x}_1$  as point  $A$ . Then  $|OA|^2 = r(\mathbf{x}_1)$ ,  $|OD|^2 = r(\mathbf{x}_2)$ ,  $|AB|^2 = r(\mathbf{x}_1 - \hat{\mathbf{x}})$  and  $|BD|^2 = r(\mathbf{x}_2 - \hat{\mathbf{x}})$  which is the desired result.

Since any convex surface remains convex after a linear transform, it follows that convex surfaces in the sinogram-space will also be convex in image-space. In fact this holds for any matrix, not merely  $\mathbf{A}$ . It follows that the bound holds for any penalty-function on the form  $r(\mathbf{x}) = \|\mathbf{L}\mathbf{x}\|_2^2$ . ■

## A2 Noise-perturbation of WLS-surface

We may determine a  $\mathbf{q}$  that satisfies  $\ell_{\text{WLS}}(\boldsymbol{\lambda} + \mathbf{p}, \mathbf{b}) = \ell_{\text{WLS}}(\boldsymbol{\lambda}, \mathbf{b} - \mathbf{q})$  by equating terms in eq. (6.5) and get  $(\lambda_i + p_i - b_i)^2/b_i = (\lambda_i - b_i + q_i)^2/(b_i - q_i)$ . Defining  $q_i = p_i c_i$  and dropping the indices, it is straightforward to show that:

$$0 = c^2 pb + c(2b(\lambda - b) + (\lambda + p - b)^2) - b(2\lambda - 2b + p)$$

Here  $c = 0$  becomes a solution in the limit  $b \rightarrow 0$ . Also, inserting  $b' = b/b = 1$ ,  $\lambda' = \lambda/b$  and  $p' = p/b$  do not change the roots of  $c$ , so we need only consider  $b = 1$ .

There are always real solutions for  $\lambda \geq 0$ , which is guaranteed since  $\lambda$  is a Poisson-estimate. We now seek the point that maximizes the root of least absolute value. This occur for  $\lambda = 0$  and  $p = 1 - \sqrt{2}$ , in which case the solutions are  $c = \pm(1 + \sqrt{2})$ .

# Base iteration

*This appendix details the basic iterative procedure used for computing large-scale SVD-problems (chapter 8) and solving linear systems (chapter 9).*

## B1 Stopping criteria

As it is often the case with numerical algorithms, one of the most difficult things of an iterative algorithm is to determine when to stop.

In principle, the base iteration consist of a loop that updates the NPSVD using alg. 7.3 p. 85. For the left residual factorization, given by eq. (7.13) with the constraint  $\hat{\mathbf{f}}_v = \mathbf{0}$ , this gives:

$$\begin{aligned} \text{(B1)} \quad & \tilde{\mathbf{s}}_v \beta^* = \mathbf{A}^H \hat{\mathbf{s}}_u - \hat{\mathbf{V}} \hat{\mathbf{f}}_u \\ \text{(B2)} \quad & \tilde{\mathbf{s}}_u \tilde{f}_u = \mathbf{A} \hat{\mathbf{s}}_v - \hat{\mathbf{s}}_u \beta \end{aligned}$$

Since we intend to use reorthogonalization, as described in sec. 7.6, the singular matrices must remain well conditioned. Therefore there are two ways to stop the iteration: (1) memory is exhausted or (2) rounding errors cause the singular matrices to loose orthogonality.

### Detecting loss of orthogonality

If we let  $\hat{\mathbf{U}}^H \hat{\mathbf{U}} = \mathbf{I} + \mathbf{M}_U$  and  $\hat{\mathbf{V}}^H \hat{\mathbf{V}} = \mathbf{I} + \mathbf{M}_V$ , then the orthogonality-level of  $\hat{\mathbf{U}}$  and  $\hat{\mathbf{V}}$  is a number,  $T$ , such that  $\|\mathbf{M}_U\|_{\max} \leq T$  and  $\|\mathbf{M}_V\|_{\max} \leq T$ .

It follows from theorem C12 p. 151 that if  $T \leq .5/k$ , then singular matrices are well-conditioned. This is less strict than the semi-orthogonality criteria discussed in sec. 7.4 p. 85, which requires  $T \leq \sqrt{\epsilon_M}$ .

One way to bound the orthogonality levels is to maintain matrices with elementwise bounds on the orthogonality, i.e.  $|\mathbf{M}_U| \leq \mathbf{O}_U$  and  $|\mathbf{M}_V| \leq \mathbf{O}_V$ . This is easiest to accomplish if only the last element of  $\hat{\mathbf{f}}_u$  is non-zero.

For this reason, the iteration should start with one step of alg. 7.3 with full re-orthogonalization. This has the additional benefit that the NPSVD is guaranteed to be expanded by at least one triplet.

### Orthogonality of left singular vectors

Consider the level of orthogonality between the search-vector  $\hat{\mathbf{s}}_{\mathbf{u}}$  and previous left singular vectors:

$$\begin{aligned} \hat{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}} &= \frac{1}{\hat{\mathbf{f}}_{\mathbf{u}}^H \hat{\mathbf{f}}_{\mathbf{u}}} \hat{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}} \hat{\mathbf{f}}_{\mathbf{u}}^H \hat{\mathbf{f}}_{\mathbf{u}} \\ \text{From eq. (8.1a)} \quad &= \frac{1}{\hat{\mathbf{f}}_{\mathbf{u}}^H \hat{\mathbf{f}}_{\mathbf{u}}} \hat{\mathbf{U}}^H (\mathbf{A} \hat{\mathbf{V}} - \hat{\mathbf{U}} \hat{\mathbf{B}} - \mathbf{G}_{\mathbf{U}}) \hat{\mathbf{f}}_{\mathbf{u}} \\ \text{From eq. (8.1b)} \quad &= \frac{1}{\hat{\mathbf{f}}_{\mathbf{u}}^H \hat{\mathbf{f}}_{\mathbf{u}}} ((\hat{\mathbf{B}} \hat{\mathbf{V}}^H + \mathbf{G}_{\mathbf{V}}^H) \hat{\mathbf{V}} - \hat{\mathbf{U}}^H \hat{\mathbf{U}} \hat{\mathbf{B}} - \hat{\mathbf{U}}^H \mathbf{G}_{\mathbf{U}}) \hat{\mathbf{f}}_{\mathbf{u}} \\ &= \frac{1}{\hat{\mathbf{f}}_{\mathbf{u}}^H \hat{\mathbf{f}}_{\mathbf{u}}} (\hat{\mathbf{B}} \mathbf{M}_{\mathbf{V}} - \mathbf{M}_{\mathbf{U}} \hat{\mathbf{B}} + \mathbf{G}_{\mathbf{V}}^H \hat{\mathbf{V}} - \hat{\mathbf{U}}^H \mathbf{G}_{\mathbf{U}}) \hat{\mathbf{f}}_{\mathbf{u}} \end{aligned}$$

Taking absolute values of the above gives:

$$\begin{aligned} \text{(B3)} \quad |\hat{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}}| &\leq \frac{1}{\hat{\mathbf{f}}_{\mathbf{u}}^H \hat{\mathbf{f}}_{\mathbf{u}}} (|\hat{\mathbf{B}}|_{\mathbf{O}_{\mathbf{V}}} |\hat{\mathbf{f}}_{\mathbf{u}}| + \mathbf{O}_{\mathbf{U}} |\hat{\mathbf{B}} \hat{\mathbf{f}}_{\mathbf{u}}| + |\mathbf{G}_{\mathbf{V}}^H \hat{\mathbf{V}} \hat{\mathbf{f}}_{\mathbf{u}}| + |\hat{\mathbf{U}}^H \mathbf{G}_{\mathbf{U}} \hat{\mathbf{f}}_{\mathbf{u}}|) \\ &= \frac{1}{|\hat{f}_{\mathbf{u}}|} (|\hat{\mathbf{B}}|_{(\mathbf{O}_{\mathbf{V}})_{[k]}} + \mathbf{O}_{\mathbf{U}} |\hat{\mathbf{B}}|_{[k]} + |\mathbf{G}_{\mathbf{V}}^H \hat{\mathbf{V}}|_{[k]} + |\hat{\mathbf{U}}^H (\mathbf{G}_{\mathbf{U}})_{[k]}|) \end{aligned}$$

Section 7.3.4 showed that elements in each of the last two terms are bounded by a constant,  $\tau_c$ . Although single elements could be larger, it is usually safe to use  $\tau_c = \tau$ .

In the rare case where this fails, it is possible to retract the last one or two iteration in the following reorthogonalization. The most common cause for this is that an optimistic value of  $\tau$  is used in the iteration.

### Orthogonality of right singular vectors

We need an analogous result for the right singular vectors. This is slightly complicated by the fact that the right search-vector is computed during the update.

$$\hat{\mathbf{s}}_{\mathbf{v}} \beta^* = \mathbf{A}^H \hat{\mathbf{s}}_{\mathbf{u}} - \hat{\mathbf{V}} \hat{\mathbf{f}}_{\mathbf{u}} + \mathbf{r}$$

where  $\mathbf{r}$  represents rounding errors. Then we have:

$$\begin{aligned} \hat{\mathbf{V}}^H \hat{\mathbf{s}}_{\mathbf{v}} &= \frac{1}{\beta^*} (\hat{\mathbf{V}}^H \mathbf{A}^H \hat{\mathbf{s}}_{\mathbf{u}} - \hat{\mathbf{V}}^H \hat{\mathbf{V}} \hat{\mathbf{f}}_{\mathbf{u}} + \hat{\mathbf{V}}^H \mathbf{r}) \\ \text{Inserting eq. (8.1a)} \quad &= \frac{1}{\beta^*} (\hat{\mathbf{B}}^H \hat{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}} + \hat{\mathbf{f}}_{\mathbf{u}} \hat{\mathbf{s}}_{\mathbf{u}}^H \hat{\mathbf{s}}_{\mathbf{u}} + \mathbf{G}_{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}} - (\mathbf{I} + \mathbf{M}_{\mathbf{V}}) \hat{\mathbf{f}}_{\mathbf{u}} + \hat{\mathbf{V}}^H \mathbf{r}) \end{aligned}$$

Using the fact that  $\hat{\mathbf{s}}_{\mathbf{u}}$  is normalized:

$$= \frac{1}{\beta^*} (\hat{\mathbf{B}}^H \hat{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}} + \mathbf{G}_{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}} - \mathbf{M}_{\mathbf{V}} \hat{\mathbf{f}}_{\mathbf{u}} + \hat{\mathbf{V}}^H \mathbf{r})$$

Taking absolute values gives the bound:

$$\text{(B4)} \quad |\hat{\mathbf{V}}^H \hat{\mathbf{s}}_{\mathbf{v}}| \leq \frac{1}{|\beta|} (|\hat{\mathbf{B}}^H| (|\hat{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}}|) + \mathbf{O}_{\mathbf{V}} |\hat{\mathbf{f}}_{\mathbf{u}}| + |\mathbf{G}_{\mathbf{U}}^H \hat{\mathbf{s}}_{\mathbf{u}}| + |\hat{\mathbf{V}}^H \mathbf{r}|)$$

### Notes on orthogonality

A few comments are in place. First of all, the bounds degrades quickly if  $|\hat{f}_{\mathbf{u}}|$  or  $|\beta|$  becomes small compared to  $\|\hat{\mathbf{B}}\|$ . However, if  $|\hat{f}_{\mathbf{u}}|$  or  $\beta$  is small, the NPSVD

has started to converge. In rough terms  $\hat{\mathbf{B}}$  will decouple if one of these entries is small enough, and “almost” decouple if it is just small.

The phenomena is well-known and has a comprehensive theory [15]. It allows us to use the loss of orthogonality to double as an efficient way to detect convergence. There is no need to provide a separate criteria.

Finally, it is possible that the bound on the search-vector is smaller than the previous level of orthogonality if  $|\hat{f}_u|$  (or  $|\beta|$ ) is large compared to  $\|\hat{\mathbf{B}}\|$ . In this case the largest value must be used.

001010  
100001  
111100

### Algorithm

Once the stopping criteria has been determined, it is relatively easy to implement the base iteration as shown in alg. B1. The orthogonality-level of  $\hat{\mathbf{U}}$  is not allowed to degrade to the same level as  $\hat{\mathbf{V}}$ . This is because  $T_v$  can not be computed until  $\beta$  is known, which requires a matrix-vector product. Thus, if the algorithm stops at this point the computation will be wasted.

The implementation used in BBTools does not even check  $T_v$ , since it can retract the last iterations to guard against an optimistic value of  $\tau$ . In many cases the bounds are pessimistic, and the retraction is not needed at all.

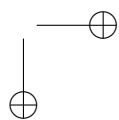
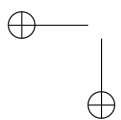
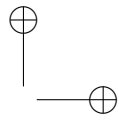
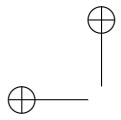
#### Algorithm B1: Base iteration

*Basic implementation of the left residual factorization. The input is an orthogonalized NPSVD updated once with alg. 7.3 with full reorthogonalization. The iteration updates the NPSVD until the singular matrices start to loose orthogonality or memory is full.*

```

function [ $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_u, f_u, k$ ] = iter( $\mathbf{A}, \hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{B}}, \hat{\mathbf{s}}_u, f_u, k, K, \tau, f_A, f_A^H, \text{varargin}$ )
     $\mathbf{O}_U = \text{ones}(K, K) * \epsilon_M$ ;    $T_u = \epsilon_M$ ;                               Initial orthogonality levels
     $\mathbf{O}_V = \text{ones}(K, K) * \epsilon_M$ ;    $T_v = \epsilon_M$ ;
    while  $k < K$                                                          Iterate until memory is full
         $\hat{\mathbf{B}}_A = |\hat{\mathbf{B}}(1:k, 1:k)|$ ;                                           Take out absolute part
         $\tilde{\mathbf{o}}_u = \mathbf{O}_U(1:k, k) * \hat{\mathbf{B}}_A(k, k) + \hat{\mathbf{B}}_A * \mathbf{O}_V(1:k, k)$ ;           Evaluate eq. (B3)
         $\tilde{\mathbf{o}}_u = (\tilde{\mathbf{o}}_u + \tau) / |f_u|$ ;
         $\mathbf{O}_U(1:k, k+1) = \tilde{\mathbf{o}}_u$ ;    $\mathbf{O}_U(k+1, 1:k) = \tilde{\mathbf{o}}_u^H$ ;           Update orthogonality matrix
         $\mathbf{O}_U(k+1, k+1) = \epsilon_M$ ;
         $T_u = \max(T_u, \max(\tilde{\mathbf{o}}_u))$ ;                                           Update orthogonality level
        if  $T_u * k > 0.05$ , break                                           Stop if orthogonality-level is low
        [ $\hat{\mathbf{s}}_v, \beta$ ] = unit( $\mathbf{A}^H \hat{\mathbf{s}}_u - \hat{\mathbf{V}}(:, k) * f_u$ );                       Search-vector by eq. (B1)
         $\tilde{\mathbf{o}}_v = \hat{\mathbf{B}}_A^H * \tilde{\mathbf{o}}_u + \mathbf{O}_V(1:k, k) * |f_u|$ ;                       Evaluate eq. (B4)
         $\tilde{\mathbf{o}}_v = (\tilde{\mathbf{o}}_v + \tau) / |\beta|$ ;
         $\mathbf{O}_V(1:k, k+1) = \tilde{\mathbf{o}}_v$ ;    $\mathbf{O}_V(k+1, 1:k) = \tilde{\mathbf{o}}_v^H$ ;           Update orthogonality matrix
         $\mathbf{O}_V(k+1, k+1) = \epsilon_M$ ;
         $T_v = \max(T_v, \max(\tilde{\mathbf{o}}_v))$ ;                                           Update orthogonality level
        if  $T_v * k > .5$ , break                                           Stop if orthogonality-level is low
         $k = k + 1$ ;    $\hat{\mathbf{U}}(:, k) = \hat{\mathbf{s}}_u$ ;    $\hat{\mathbf{V}}(:, k) = \hat{\mathbf{s}}_v$ ;           Update singular matrices
         $\hat{\mathbf{B}}(k, 1:k) = 0$ ;    $\hat{\mathbf{B}}(1:k, k) = 0$ ;                               Update coupling matrix
         $\hat{\mathbf{B}}(k, k-1) = (f_u)^*$ ;    $\hat{\mathbf{B}}(k, k) = \beta$ ;
        [ $\hat{\mathbf{s}}_u, f_u$ ] = unit( $\mathbf{A} * \hat{\mathbf{s}}_v - \hat{\mathbf{s}}_u * \beta$ );                       Search-vector from eq. (B2)

```



# Matrix analysis

*This appendix contains useful results and definitions from matrix analysis that are used in the matrix algorithms in part III. Although some results are proved here we strive to give references where possible.*

*Many perturbation results found in the literature are given in terms of angles between subspaces. In real programs it is normally more convenient to use inner products and norms. This is the emphasis of the forms given here.*

## C1 Perturbation theory

Numerical methods rarely give exact result, but perturbation theory gives bounds that we can use to guide the algorithms. The most important perturbation result about the SVD is given by Weyl's theorem.

**Theorem C1:** (Weyl) *The matrices  $\mathbf{A}, \mathbf{E} \in \mathbb{C}^{m \times n}$  satisfy  $|\sigma_k(\mathbf{A} + \mathbf{E}) - \sigma_k(\mathbf{A})| \leq \|\mathbf{E}\|_2$ .*

PROOF See [69]. ■

A useful consequence of Weyl's theorem is the following corollary:

**Corollary C2:** *Let  $\mathbf{A}$  be the following block matrix:*

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{bmatrix}$$

*Then  $\|\mathbf{A}\|_2 \leq \max\{\|\mathbf{A}_{1,1}\|_2, \|\mathbf{A}_{2,2}\|_2\} + \max\{\|\mathbf{A}_{1,2}\|_2, \|\mathbf{A}_{2,1}\|_2\}$ .*

There are also relative versions of Weyl's theorem. We will use the most common bound, but others have been found recently [51].

**Theorem C3:** (Ostrowski) *Assume  $\tilde{\mathbf{A}} = \mathbf{D}_1^H \mathbf{A} \mathbf{D}_2$  and  $\mathbf{A} \in \mathbb{C}^{m \times n}$ , where  $\mathbf{D}_1$  and  $\mathbf{D}_2$  are square. Let  $\sigma_k$  and  $\tilde{\sigma}_k$  the  $k$ 'th singular value of  $\mathbf{A}$  and  $\tilde{\mathbf{A}}$ .*

*Further, define  $c_M = \sigma_1(\mathbf{D}_1)\sigma_1(\mathbf{D}_2)$ , the product of the largest singular values, and  $c_m = \sigma_m(\mathbf{D}_1)\sigma_n(\mathbf{D}_2)$ , the product of the smallest singular values. Then we have:*

$$c_m \sigma_i \leq \tilde{\sigma}_i \leq c_M \sigma_i$$

PROOF See [51]. ■

## C2 Unitarily invariant norms

A norm,  $\|\cdot\|$ , is unitarily invariant if  $\|\mathbf{A}\| = \|\mathbf{U}^H \mathbf{A} \mathbf{V}\|$  for all unitary matrices  $\mathbf{U}$  and  $\mathbf{V}$ . Many stable algorithms are based on orthogonal transformations, and this puts unitarily invariant norms in a central position for measuring rounding errors. This is true even though they are not usually absolute<sup>1</sup>.

<sup>1</sup>A norm is absolute if  $\|\mathbf{A}\| = \|(|\mathbf{A}|)\|$ . The matrix 2-norm is not absolute.

Since the SVD reduces any matrix to diagonal form, it is convenient to define  $\boldsymbol{\sigma}(\mathbf{A})$  as a vector containing the singular values of  $\mathbf{A}$ . With this notation we may write the two most important norms as  $\|\mathbf{A}\|_2 = \|\boldsymbol{\sigma}(\mathbf{A})\|_\infty$  and  $\|\mathbf{A}\|_F = \|\boldsymbol{\sigma}(\mathbf{A})\|_2$ . However, the following theorem, due to John Von Neumann, shows that the family is infinite.

**Theorem C4: (Von Neumann's norm theorem)** *The function  $\|\mathbf{A}\| = \|\boldsymbol{\sigma}(\mathbf{A})\|_V$  is a unitarily invariant matrix-norm if, and only if,  $\|\mathbf{x}\|_V$  is an absolute permutation invariant vector-norm.*

PROOF See [70]. ■

This shows that a result that holds for any unitarily invariant norm is quite strong. One way to prove such results is to use the following theorem.

**Theorem C5: (Ky Fan's dominance theorem)** *Let  $\mathbf{A}$  and  $\mathbf{B}$  have equal size. Then  $\|\mathbf{A}\| \leq \|\mathbf{B}\|$  for all unitarily invariant norms if, and only if,  $\sum_{i=1}^k \sigma_i(\mathbf{A}) \leq \sum_{i=1}^k \sigma_i(\mathbf{B})$  for  $k = 1, 2, \dots$ .*

PROOF See [70]. ■

The dominance theorem is sometimes impractical to use directly, but a few theorems take some of the labor away.

**Theorem C6: (Padded matrix dominance)** *Let  $\mathbf{A}$  be any matrix. Then any unitarily invariant norm satisfies  $\|[\mathbf{A}, \mathbf{0}]\| \leq \|[\mathbf{A}, \mathbf{B}]\|$ .*

PROOF We need only show the case where  $\mathbf{B} = \mathbf{b}$  is a single column, since the general result then follows by adding a single column at a time. Now set  $\mathbf{X} = [\mathbf{A}, \mathbf{b}]$  and form

$$\mathbf{X}^H \mathbf{X} = \begin{bmatrix} \mathbf{A}^H \mathbf{A} & \mathbf{A}^H \mathbf{b} \\ \mathbf{b}^H \mathbf{A} & \mathbf{b}^H \mathbf{b} \end{bmatrix}.$$

By the Cauchy interlace theorem [27], the eigenvalues of  $\mathbf{A}^H \mathbf{A}$  interlaces those of  $\mathbf{X}^H \mathbf{X}$ . For  $\mathbf{b} = \mathbf{0}$  the eigenvalues of  $\mathbf{X}^H \mathbf{X}$  are the eigenvalues of  $\mathbf{A}^H \mathbf{A}$  and a zero. Since the eigenvalues of  $\mathbf{X}^H \mathbf{X}$  are the squared singular values of  $\mathbf{X}$ , we have  $\sigma_i([\mathbf{A}, \mathbf{0}]) \leq \sigma_i([\mathbf{A}, \mathbf{b}])$  and the dominance property is satisfied. ■

**Theorem C7: (Projection dominance)** *Let  $\mathbf{P}$  be an idempotent projection, i.e.  $\mathbf{P} = \mathbf{P}\mathbf{P}$ . Then  $\|\mathbf{P}\mathbf{A}\| \leq \|\mathbf{A}\|$  in any unitarily invariant norm.*

PROOF The singular values of an idempotent matrix,  $\mathbf{P}$ , are 1 or 0. The theorem now follows from Ostrowski's theorem C3 and C5. ■

## Normalized norms

We are interested in norms that bound the error of individual components so that  $\|\mathbf{A}\| \geq \max_{i,j} |A_{i,j}|$ . To make this inequality sharp, we assume the norm satisfies  $\|\mathbf{A}\| = |c|$  if  $\mathbf{A}$  consist of a single non-zero entry  $c$ . This is equivalent to requiring  $\|\mathbf{A}\| = \|\mathbf{A}\|_2$  if  $\text{rank}(\mathbf{A}) = 1$ .

Ky Fan's dominance theorem implies that the 2-norm is minimal in the sense that  $\|\mathbf{X}\|_2 \leq \|\mathbf{X}\|$ , which follows from inserting  $\mathbf{A} = \text{diag}(\|\mathbf{X}\|_2, 0, \dots)$  and  $\mathbf{B} = \mathbf{X}$ .



The Ky Fan norm, defined as  $\|\mathbf{A}\|_{\text{KF}} = \|\boldsymbol{\sigma}(\mathbf{A})\|_1$ , is the other extreme. If we set  $\mathbf{A} = \mathbf{X}$  and  $\mathbf{B} = \text{diag}(\|\mathbf{X}\|_{\text{KF}}, 0, \dots)$  and apply the dominance theorem we get  $\|\mathbf{X}\| \leq \|\mathbf{X}\|_{\text{KF}}$ . If we combine these observations we get the following result.

$$(C1) \quad \|\mathbf{A}\|_2 \leq \|\mathbf{A}\| \leq \|\mathbf{A}\|_{\text{KF}}$$

### C3 Norm inequalities

All norms can be bounded in terms of each other, and it is handy to have a list of these bounds ready. In addition to the usual norms,  $\|\cdot\|_1$ ,  $\|\cdot\|_2$ ,  $\|\cdot\|_\infty$ , and  $\|\cdot\|_F$ , we also need the max-norm:

$$(C2) \quad \|\mathbf{A}\|_{\text{max}} = \max_{i,j} |\mathbf{A}_{(i,j)}|$$

The max-norm is not consistent, i.e. the inequality  $\|\mathbf{AB}\| \leq \|\mathbf{A}\| \cdot \|\mathbf{B}\|$  may not be satisfied. However, the max-norm gives a component-wise error which makes it a natural choice in some situations. This thesis uses it mainly to measure the orthogonality level or near-orthogonal matrices.

Table C.1 give constants for the norms of used in this thesis for a matrix  $\mathbf{A} \in \mathbb{C}^{m \times n}$ . For convenience we also define  $d = \min\{m, n\}$ . The upper bound is found starting on the left, while the lower bound is found from the top. For instance we find  $\|\mathbf{A}\|_1 / \sqrt{m} \leq \|\mathbf{A}\|_2 \leq \sqrt{n} \|\mathbf{A}\|_1$  in two lookups.

**Table C.1:**  
Constants that bounds the norm on the left in terms of the norm on the top. The matrix  $\mathbf{A}$  is  $m \times n$ , and  $d = \min\{m, n\}$ .

	$\ \mathbf{A}\ _{\text{max}}$	$\ \mathbf{A}\ _1$	$\ \mathbf{A}\ _\infty$	$\ \mathbf{A}\ _2$	$\ \mathbf{A}\ _F$	$\ \mathbf{A}\ _{\text{KF}}$
$\ \mathbf{A}\ _{\text{max}}$		1	1	1	1	1
$\ \mathbf{A}\ _1$	$m$		$m$	$\sqrt{m}$	$\sqrt{m}$	$\sqrt{m}$
$\ \mathbf{A}\ _\infty$	$n$	$n$		$\sqrt{n}$	$\sqrt{n}$	$\sqrt{n}$
$\ \mathbf{A}\ _2$	$\sqrt{mn}$	$\sqrt{n}$	$\sqrt{m}$		1	1
$\ \mathbf{A}\ _F$	$\sqrt{mn}$	$\sqrt{n}$	$\sqrt{m}$	$\sqrt{d}$		1
$\ \mathbf{A}\ _{\text{KF}}$	$\sqrt{mnd}$	$\sqrt{nd}$	$\sqrt{md}$	$d$	$\sqrt{d}$	

The constants in the table are attained for some matrices. For instance the matrix  $\mathbf{A}_{(i,j)} = e^{2\pi i j / \max\{m, n\}}$  satisfies  $\|\mathbf{A}\|_{\text{KF}} = \sqrt{mnd} \|\mathbf{A}\|_{\text{max}}$ . It is not always possible to attain equality if  $\mathbf{A}$  is restricted to be real (e.g.  $3 \times 3$ ). It is still asymptotically sharp, and it can sometimes be reached. If, for example,  $\mathbf{H} = [\mathbf{A}, \mathbf{B}]$  is a Hadamard matrix then equality is attained by  $\mathbf{A}$ .

Finally we also need the following bounds [27, 69]:

$$(C3) \quad \|\mathbf{A}\|_2 \leq \|(|\mathbf{A}|)\|_2$$

$$(C4) \quad \|\mathbf{A}\|_2 \leq \sqrt{\|\mathbf{A}\|_1 \|\mathbf{A}\|_\infty}$$

$$(C5) \quad |\mathbf{x}^H \mathbf{y}| \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$$

We refer to eq. (C5) as the Cauchy inequality, although some texts refer to it as the Cauchy-Schwartz inequality.

### C4 Matrix approximations

Any good textbook on linear algebra discusses the least-squares problem where we find an approximate solution to  $\mathbf{Ax} \approx \mathbf{b}$  by minimizing  $\|\mathbf{Ax} - \mathbf{b}\|_2$ . It

turns out that the notion of “least-squares” generalizes naturally from vectors to matrices. This result is established in the following theorem.

**Theorem C8: (Least-squares approximation)** *Let  $\mathbf{A} \in \mathbb{C}^{m \times n}$  and  $\mathbf{B} \in \mathbb{C}^{m \times p}$  be arbitrary matrices. Then  $\mathbf{X} = \mathbf{A}^+ \mathbf{B}$  minimizes  $\|\mathbf{A}\mathbf{X} - \mathbf{B}\|$ , and is the unique choice that minimizes  $\|\mathbf{X}\|$ , where  $\|\cdot\|$  is any unitarily invariant norm.*

**PROOF** Let  $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H$  be a full SVD of  $\mathbf{A}$ , and define  $\mathbf{X} = \mathbf{A}^+ \mathbf{B} + \mathbf{D}$  where  $\mathbf{D} \in \mathbb{C}^{n \times p}$ . For  $k = \text{rank}(\mathbf{A})$  we have:

$$\begin{aligned} \|\mathbf{A}\mathbf{X} - \mathbf{B}\| &= \|\mathbf{A}\mathbf{A}^+ \mathbf{B} - \mathbf{B} + \mathbf{A}\mathbf{D}\| \\ &= \|(\mathbf{A}\mathbf{A}^+ - \mathbf{I})\mathbf{B} + \mathbf{A}\mathbf{D}\| \\ &= \|((\mathbf{U}\mathbf{\Sigma}\mathbf{V}^H)(\mathbf{V}\mathbf{\Sigma}^+ \mathbf{U}^H) - \mathbf{I})\mathbf{B} + \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H \mathbf{D}\| \\ &= \|(\mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^+ \mathbf{U}^H - \mathbf{U}\mathbf{U}^H)\mathbf{B} + \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H \mathbf{D}\| \\ &= \|(\mathbf{\Sigma}\mathbf{\Sigma}^+ - \mathbf{I})\mathbf{U}^H \mathbf{B} + \mathbf{\Sigma}\mathbf{V}^H \mathbf{D}\| \\ &= \left\| \begin{bmatrix} \mathbf{0} & \\ & -\mathbf{I} \end{bmatrix} \mathbf{U}^H \mathbf{B} + \begin{bmatrix} \mathbf{\Sigma}_k & \\ & \mathbf{0} \end{bmatrix} \mathbf{V}^H \mathbf{D} \right\| \end{aligned}$$

(unitary invariance)

where  $\mathbf{\Sigma}_k = \text{diag}(\sigma_1, \dots, \sigma_k)$ .

Since the rows of the two terms do not overlap, it follows from the padded dominance theorem C6, that the first  $k$  rows of  $\mathbf{V}^H \mathbf{D}$  must be zero. We also have:

$$\|\mathbf{X}\| = \|\mathbf{V}\mathbf{\Sigma}^+ \mathbf{U}^H \mathbf{B} + \mathbf{D}\| = \|\mathbf{\Sigma}^+ \mathbf{U}^H \mathbf{B} + \mathbf{V}^H \mathbf{D}\|$$

Only the first  $k$  rows of the first term can be non-zero, and we have just seen that these must be zero in the second term for  $\mathbf{X}$  to minimize  $\|\mathbf{A}\mathbf{X} - \mathbf{B}\|$ . As before the rows do not overlap, so the remaining rows of  $\mathbf{V}^H \mathbf{D}$  must be zero to minimize  $\|\mathbf{X}\|$ . ■

The following result is well-known, but is not usually stated in full generality.

**Theorem C9: (Schmidt-Mirsky low-rank approximation)** *Let the SVD of  $\mathbf{A} \in \mathbb{C}^{m \times n}$  be given by  $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H$  and let  $\|\cdot\|$  be any unitarily invariant norm. Then the matrix  $\mathbf{B} = \sum_{i=1}^k \sigma_{(i,i)} \mathbf{U}_{(:,i)} \mathbf{V}_{(:,i)}^H$  minimizes  $\|\mathbf{A} - \mathbf{B}\|$  among all matrices with  $\text{rank}(\mathbf{B}) = k \leq \text{rank}(\mathbf{A})$ .*

**PROOF** A number of texts include a proof for special cases. A particularly readable geometric argument that covers the 2-norm is given in [73]. A proof for the general case can be found in [70]. ■

## C5 Near-orthogonal matrices

It is useful to collect a few facts about matrices on the form  $\hat{\mathbf{Q}}^H \hat{\mathbf{Q}} \approx \mathbf{I}$ . To this end we consider the matrix  $\mathbf{Q}^H \mathbf{Q} = \mathbf{I} + \mathbf{M}_{\mathbf{Q}}$ . The *orthogonality level* of the columns in  $\mathbf{Q}$  is then defined as the scalar  $T = \|\mathbf{M}_{\mathbf{Q}}\|_{\max}$ . A key property of near-orthogonal matrices is that their singular values are near 1.

**Theorem C10: (Singular values of a near-orthogonal matrix)** *Let the columns of  $\mathbf{Q} \in \mathbb{C}^{m \times n}$  be orthogonal at level  $T$ , i.e.  $\mathbf{Q}^H \mathbf{Q} = \mathbf{I} + \mathbf{E}$  where  $\|\mathbf{E}\|_{\max} \leq T$ . If  $nT \leq 1$ , then  $\sqrt{1 - nT} \leq \sigma_i(\mathbf{Q}) \leq \sqrt{1 + nT}$  for all  $i$ .*

**PROOF** From table C.1 we have  $\|\mathbf{E}\|_2 \leq n\|\mathbf{E}\|_{\max} \leq nT$ . Weyl’s theorem then gives the bound  $1 - nT \leq \sigma_i(\mathbf{I} + \mathbf{E}) = \sigma_i(\mathbf{Q}^H \mathbf{Q}) \leq 1 + nT$ . But  $\sigma_i(\mathbf{Q}) = \sqrt{\sigma_i(\mathbf{Q}^H \mathbf{Q})}$ , which establishes the theorem. ■

A near-orthogonal matrix is always well-conditioned. We start with a lemma.

**Lemma C11:** *If  $|x| \leq \kappa < 1$  are real scalars, then  $(1+x)^2 \leq (1+x)/(1-x) \leq (1+x)^2/(1-\kappa^2)$ .*

**PROOF** Multiplying through with  $1+x$  we get:

$$\frac{1+x}{1-x} = \frac{(1+x)^2}{1-x^2}$$

The lemma follows since  $|x| \leq \kappa < 1$  implies that  $1 - \kappa^2 \leq 1 - x^2 \leq 1$ . ■

**Theorem C12: (Condition of near-orthogonal matrix)** *Let the columns of  $\mathbf{Q} \in \mathbb{C}^{m \times n}$  be orthogonal at level  $T$ , i.e.  $\mathbf{Q}^H \mathbf{Q} = \mathbf{I} + \mathbf{E}$  where  $\|\mathbf{E}\|_{\max} \leq T$ . If  $nT \leq \kappa < 1$ , then  $\text{cond}(\mathbf{Q}) \leq (1+\kappa)/\sqrt{1-\kappa^2}$ .*

**PROOF** Since the condition of  $\mathbf{Q}$  is given by  $\text{cond}(\mathbf{Q}) = \max_i \sigma_i(\mathbf{Q}) / \min_i \sigma_i(\mathbf{Q})$ , theorem C10 states that  $\text{cond}(\mathbf{Q}) \leq \sqrt{(1+nT)/(1-nT)}$ . The theorem follows from inserting  $x = nT$  in lemma C11. ■

For instance, if  $\kappa \leq 1/2$ , then  $\text{cond}(\mathbf{Q}) \leq \sqrt{3}$ .

## C6 Triplet orthogonality

One of the big hurdles in creating numerical software for the SVD (and many other problems), is the need to keep the triplets orthogonal. In order get meaningful results, many iterative methods must sacrifice efficiency. The time spent during orthogonalization may dominate the whole computation

Explicit reorthogonalization becomes intolerable for problems that are too large to contain a solution in memory. In this case reorthogonalization requires disk access which severely hampers the efficiency of the algorithm.

To circumvent this we will rely on *implicit orthogonality*. The theorems derived in this section tells us that numerical triplets will automatically be (numerically) orthogonal under certain conditions.

We consider numerical singular triplets  $(\hat{\mathbf{u}}_i, \hat{\mathbf{v}}_i, \hat{\sigma}_i)$  of the form:

$$(C6) \quad \mathbf{A} \hat{\mathbf{v}}_i = \hat{\sigma}_i \hat{\mathbf{u}}_i + \check{\mathbf{u}}_i \quad \mathbf{A}^H \hat{\mathbf{u}}_i = \hat{\sigma}_i^* \hat{\mathbf{v}}_i + \check{\mathbf{v}}_i$$

It is assumed that all triplets have been normalized, i.e.  $1 - \epsilon_o \leq \|\hat{\mathbf{u}}_i\|, \|\hat{\mathbf{v}}_i\| \leq 1 + \epsilon_o$  where  $\epsilon_o$  is on the order of machine-precision.

The results are derived by expressing the inner products in terms of the errors.

**Lemma C13:** *A pair of triplets, as given by eq. (C6), where  $|\hat{\sigma}_i| \neq |\hat{\sigma}_j|$  satisfies the relations:*

$$\begin{aligned} \hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i &= \frac{\hat{\sigma}_i (\check{\mathbf{u}}_j^H \hat{\mathbf{u}}_i - \hat{\mathbf{v}}_j^H \check{\mathbf{v}}_i) + \hat{\sigma}_j^* (\check{\mathbf{v}}_j^H \hat{\mathbf{v}}_i - \hat{\mathbf{u}}_j^H \check{\mathbf{u}}_i)}{|\hat{\sigma}_i|^2 - |\hat{\sigma}_j|^2} \\ \hat{\mathbf{u}}_j^H \hat{\mathbf{u}}_i &= \frac{\hat{\sigma}_i^* (\check{\mathbf{v}}_j^H \hat{\mathbf{v}}_i - \hat{\mathbf{u}}_j^H \check{\mathbf{u}}_i) + \hat{\sigma}_j (\check{\mathbf{u}}_j^H \hat{\mathbf{u}}_i - \hat{\mathbf{v}}_j^H \check{\mathbf{v}}_i)}{|\hat{\sigma}_i|^2 - |\hat{\sigma}_j|^2} \end{aligned}$$

PROOF Consider the entity  $\hat{\mathbf{v}}_j^H \mathbf{A}^H \mathbf{A} \hat{\mathbf{v}}_i$ :

$$\begin{aligned}\hat{\mathbf{v}}_j^H \mathbf{A}^H \mathbf{A} \hat{\mathbf{v}}_i &= \hat{\mathbf{v}}_j^H \mathbf{A}^H (\hat{\sigma}_i \hat{\mathbf{u}}_i + \check{\mathbf{u}}_i) \\ &= \hat{\sigma}_i \hat{\mathbf{v}}_j^H (\mathbf{A}^H \hat{\mathbf{u}}_i) + (\hat{\mathbf{v}}_j^H \mathbf{A}^H) \check{\mathbf{u}}_i \\ &= \hat{\sigma}_i \hat{\mathbf{v}}_j^H (\hat{\sigma}_i^* \hat{\mathbf{v}}_i + \check{\mathbf{v}}_i) + (\hat{\sigma}_j \hat{\mathbf{u}}_j + \check{\mathbf{u}}_j)^H \check{\mathbf{u}}_i \\ &= |\hat{\sigma}_i|^2 (\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i) + \hat{\sigma}_i (\hat{\mathbf{v}}_j^H \check{\mathbf{v}}_i) + \hat{\sigma}_j^* (\hat{\mathbf{u}}_j^H \check{\mathbf{u}}_i) + (\check{\mathbf{u}}_j^H \check{\mathbf{u}}_i)\end{aligned}$$

Exchanging  $\hat{\mathbf{v}}_i$  with  $\hat{\mathbf{v}}_j$  and taking the Hermitian gives another expression:

$$\hat{\mathbf{v}}_j^H \mathbf{A}^H \mathbf{A} \hat{\mathbf{v}}_i = |\hat{\sigma}_j|^2 (\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i) + \hat{\sigma}_j^* (\check{\mathbf{v}}_j^H \hat{\mathbf{v}}_i) + \hat{\sigma}_i (\check{\mathbf{u}}_j^H \hat{\mathbf{u}}_i) + (\check{\mathbf{u}}_j^H \check{\mathbf{u}}_i)$$

Isolating  $\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i$  from these equations gives the first equation in the lemma. An analogous argument on  $\hat{\mathbf{u}}_j^H \mathbf{A} \mathbf{A}^H \hat{\mathbf{u}}_i$  gives the other equation. ■

The following theorem asserts that accurate triplets are orthogonal, as long as their singular values differ slightly.

**Theorem C14: (Mutual triplet orthogonality)** *Assume a set of normalized numerical triplets  $(\hat{\mathbf{u}}_i, \hat{\mathbf{v}}_i, \hat{\sigma}_i)$  satisfies  $\|\mathbf{A} \hat{\mathbf{v}}_i - \hat{\sigma}_i \hat{\mathbf{u}}_i\| \leq t_i$  and  $\|\mathbf{A}^H \hat{\mathbf{u}}_i - \hat{\sigma}_i^* \hat{\mathbf{v}}_i\| \leq t_i$ . Then the orthogonality of a pair of triplets, where  $|\hat{\sigma}_i| \neq |\hat{\sigma}_j|$ , is bounded by:*

$$\max\{|\hat{\mathbf{v}}_i^H \hat{\mathbf{v}}_j|, |\hat{\mathbf{u}}_i^H \hat{\mathbf{u}}_j|\} \leq \frac{t_i + t_j}{||\hat{\sigma}_i| - |\hat{\sigma}_j||} (1 + \epsilon_o)$$

PROOF First note that  $|\hat{\sigma}_i|^2 - |\hat{\sigma}_j|^2| = ||\hat{\sigma}_i| - |\hat{\sigma}_j|| \cdot (|\hat{\sigma}_i| + |\hat{\sigma}_j|)$ . This, the Cauchy inequality, and lemma C13 leads to:

$$\begin{aligned}|\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i| &\leq \frac{|\hat{\sigma}_i| \cdot (\|\check{\mathbf{u}}_j\| + \|\check{\mathbf{v}}_i\|) + |\hat{\sigma}_j| \cdot (\|\check{\mathbf{v}}_j\| + \|\check{\mathbf{u}}_i\|)}{||\hat{\sigma}_i| - |\hat{\sigma}_j|| \cdot (|\hat{\sigma}_i| + |\hat{\sigma}_j|)} (1 + \epsilon_o) \\ |\hat{\mathbf{u}}_j^H \hat{\mathbf{u}}_i| &\leq \frac{|\hat{\sigma}_i| \cdot (\|\check{\mathbf{v}}_j\| + \|\check{\mathbf{u}}_i\|) + |\hat{\sigma}_j| \cdot (\|\check{\mathbf{u}}_j\| + \|\check{\mathbf{v}}_i\|)}{||\hat{\sigma}_i| - |\hat{\sigma}_j|| \cdot (|\hat{\sigma}_i| + |\hat{\sigma}_j|)} (1 + \epsilon_o)\end{aligned}$$

Substituting  $t_i$  and  $t_j$  into these equations completes the proof. ■

If there is just a small relative difference in the singular values, then the orthogonality is determined by the accuracy relative to the largest singular value. To make this clear, assume that  $|\hat{\sigma}_i| > |\hat{\sigma}_j|$  and rewrite the relation as follows:

$$\max\{|\hat{\mathbf{v}}_i^H \hat{\mathbf{v}}_j|, |\hat{\mathbf{u}}_i^H \hat{\mathbf{u}}_j|\} \leq \frac{t_i + t_j}{|\hat{\sigma}_i|} \left( \frac{1 + \epsilon_o}{1 - |\hat{\sigma}_j|/|\hat{\sigma}_i|} \right)$$

For instance, if  $|\hat{\sigma}_i| \geq 1.01|\hat{\sigma}_j|$ , then the last term is at most  $100(1 + \epsilon_o)$ .

The following result treats the case where  $|\hat{\sigma}_i| \approx |\hat{\sigma}_j|$ . It shows that keeping one set of singular triplets orthogonal is good enough for the large singular values.

**Theorem C15: (One-sided triplet orthogonality)** *Assume the normalized numerical triplets  $(\hat{\mathbf{u}}_i, \hat{\mathbf{v}}_i, \hat{\sigma}_i)$  satisfies  $\|\mathbf{A} \hat{\mathbf{v}}_i - \hat{\sigma}_i \hat{\mathbf{u}}_i\| \leq t_i$  and  $\|\mathbf{A}^H \hat{\mathbf{u}}_i - \hat{\sigma}_i^* \hat{\mathbf{v}}_i\| \leq t_i$ . Then we have the bounds:*

$$\begin{aligned}|\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i| &\leq |\hat{\mathbf{u}}_j^H \hat{\mathbf{u}}_i| + \frac{t_i + t_j}{\min\{|\hat{\sigma}_i|, |\hat{\sigma}_j|\}} (1 + \epsilon_o) \\ |\hat{\mathbf{u}}_j^H \hat{\mathbf{u}}_i| &\leq |\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i| + \frac{t_i + t_j}{\min\{|\hat{\sigma}_i|, |\hat{\sigma}_j|\}} (1 + \epsilon_o)\end{aligned}$$

PROOF

We first assume  $|\hat{\sigma}_i| < |\hat{\sigma}_j|$ , which only leaves equality since the triplets can be exchanged. We then rewrite lemma C13 in the form:

$$(C8) \quad \hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i = \hat{\sigma}_i c_1 + \hat{\sigma}_j^* c_2$$

$$(C9) \quad \hat{\mathbf{u}}_j^H \hat{\mathbf{u}}_i = \hat{\sigma}_i^* c_2 + \hat{\sigma}_j c_1$$

Here we have defined the values  $c_1 = (\check{\mathbf{u}}_j^H \hat{\mathbf{u}}_i - \hat{\mathbf{v}}_j^H \check{\mathbf{v}}_i) / (|\hat{\sigma}_i|^2 - |\hat{\sigma}_j|^2)$  and  $c_2 = (\check{\mathbf{v}}_j^H \hat{\mathbf{v}}_i - \hat{\mathbf{u}}_j^H \check{\mathbf{u}}_i) / (|\hat{\sigma}_i|^2 - |\hat{\sigma}_j|^2)$ .

Isolating  $c_1$  from eq. (C8) and inserting into eq. (C9) gives:

$$\begin{aligned} \hat{\mathbf{u}}_j^H \hat{\mathbf{u}}_i &= \hat{\sigma}_i^* c_2 + \frac{\hat{\sigma}_j}{\hat{\sigma}_i} (\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i - \hat{\sigma}_j^* c_2) \\ &= \frac{c_2}{\hat{\sigma}_i} (|\hat{\sigma}_i|^2 - |\hat{\sigma}_j|^2) + \frac{\hat{\sigma}_i}{\hat{\sigma}_j} (\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i) \\ &= \frac{\check{\mathbf{v}}_j^H \hat{\mathbf{v}}_i - \hat{\mathbf{u}}_j^H \check{\mathbf{u}}_i}{\hat{\sigma}_i} + \frac{\hat{\sigma}_i}{\hat{\sigma}_j} (\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i) \end{aligned}$$

Similarly, isolating  $c_2$  from eq. (C9) and inserting in eq. (C8) gives:

$$\hat{\mathbf{v}}_j^H \hat{\mathbf{v}}_i = \frac{\check{\mathbf{u}}_j^H \hat{\mathbf{u}}_i - \hat{\mathbf{v}}_j^H \check{\mathbf{v}}_i}{\hat{\sigma}_i^*} + \frac{\hat{\sigma}_i^*}{\hat{\sigma}_j^*} (\hat{\mathbf{u}}_j^H \hat{\mathbf{u}}_i)$$

Inserting the bounds required by the theorem completes the case where  $|\hat{\sigma}_i| \neq |\hat{\sigma}_j|$ . In the case of equality we simply multiply  $\sigma_i$  by a factor,  $c$ , which will increase  $t_i$  and  $t_j$  by an arbitrarily small amount. In the limit  $c \rightarrow 1$  we obtain the same relation. ■

## C7 Accuracy of singular values

To control the accuracy of the computations we need to bound the accuracy of the singular values in terms of quantities that can either be computed or bounded. In practice this means that it should be expressed in terms of the norms of the residuals matrices,  $\|\mathbf{E}_U\|$  and  $\|\mathbf{E}_V\|$ , and the orthogonality levels,  $\|\mathbf{M}_U\|$  and  $\|\mathbf{M}_V\|$ .

To this end we start with the following lemma.

**Lemma C16:** *Let  $\hat{\mathbf{U}} \in \mathbb{C}^{m \times k}$ ,  $\hat{\mathbf{V}} \in \mathbb{C}^{n \times k}$ ,  $k \leq \min\{m, n\}$ , where  $\hat{\mathbf{U}}^H \hat{\mathbf{U}} = \mathbf{I} + \mathbf{M}_U$  and  $\hat{\mathbf{V}}^H \hat{\mathbf{V}} = \mathbf{I} + \mathbf{M}_V$ . Also, let  $\hat{\Sigma}$  be a diagonal matrix with positive real entries that satisfies:*

$$\mathbf{A} \hat{\mathbf{V}} = \hat{\mathbf{U}} \hat{\Sigma} + \mathbf{E}_U \quad \mathbf{A}^H \hat{\mathbf{U}} = \hat{\mathbf{V}} \hat{\Sigma}^H + \mathbf{E}_V$$

*Then there exists matrices  $\check{\mathbf{U}} = [\hat{\mathbf{U}}, \hat{\mathbf{U}}^\circ] \in \mathbb{C}^{m \times m}$  and  $\check{\mathbf{V}} = [\hat{\mathbf{V}}, \hat{\mathbf{V}}^\circ] \in \mathbb{C}^{n \times n}$  where  $(\hat{\mathbf{U}}^\circ)^H \hat{\mathbf{U}}^\circ = \mathbf{I}$  and  $(\hat{\mathbf{V}}^\circ)^H \hat{\mathbf{V}}^\circ = \mathbf{I}$  and satisfies:*

$$(C10) \quad \hat{\mathbf{U}}^H \hat{\mathbf{U}}^\circ = \mathbf{0} \quad \hat{\mathbf{V}}^H \hat{\mathbf{V}}^\circ = \mathbf{0} \quad (\hat{\mathbf{U}}^\circ)^H \mathbf{A} \hat{\mathbf{V}}^\circ = \hat{\Sigma}^\circ$$

*Furthermore, the product  $\mathbf{C} = \check{\mathbf{U}}^H \mathbf{A} \check{\mathbf{V}}$  is given by:*

$$(C11) \quad \mathbf{C} = \begin{bmatrix} \hat{\mathbf{U}}^H \mathbf{A} \hat{\mathbf{V}} & \mathbf{E}_V^H \hat{\mathbf{V}}^\circ \\ (\hat{\mathbf{U}}^\circ)^H \mathbf{E}_U & \hat{\Sigma}^\circ \end{bmatrix}$$

$$(C12) \quad = \begin{bmatrix} \mathbf{I} + \mathbf{M}_U & \\ & \mathbf{I} \end{bmatrix} \begin{bmatrix} \hat{\Sigma} \\ \hat{\Sigma}^\circ \end{bmatrix} + \mathbf{G}_U, \quad \mathbf{G}_U = \begin{bmatrix} \hat{\mathbf{U}}^H \mathbf{E}_U & \mathbf{E}_V^H \hat{\mathbf{V}}^\circ \\ (\hat{\mathbf{U}}^\circ)^H \mathbf{E}_U & \mathbf{0} \end{bmatrix}$$

$$(C13) \quad = \begin{bmatrix} \hat{\Sigma} \\ \hat{\Sigma}^\circ \end{bmatrix} \begin{bmatrix} \mathbf{I} + \mathbf{M}_V & \\ & \mathbf{I} \end{bmatrix} + \mathbf{G}_V, \quad \mathbf{G}_V = \begin{bmatrix} \mathbf{E}_V^H \hat{\mathbf{V}} & \mathbf{E}_V^H \hat{\mathbf{V}}^\circ \\ (\hat{\mathbf{U}}^\circ)^H \mathbf{E}_U & \mathbf{0} \end{bmatrix}$$

PROOF First choose matrices  $\hat{\mathbf{U}}_1^H \hat{\mathbf{U}}_1 = \mathbf{I}$  and  $\hat{\mathbf{V}}_1^H \hat{\mathbf{V}}_1 = \mathbf{I}$  so  $\hat{\mathbf{U}}^H \hat{\mathbf{U}}_1 = \mathbf{0}$  and  $\hat{\mathbf{V}}^H \hat{\mathbf{V}}_1 = \mathbf{0}$ . This can be done e.g. using a QR-decomposition. If  $\hat{\mathbf{U}}_2 \hat{\mathbf{\Sigma}}^\circ \hat{\mathbf{V}}_2^H = \mathbf{X}$  is an SVD of  $\mathbf{X} = \hat{\mathbf{U}}_1^H \mathbf{A} \hat{\mathbf{V}}_1$ , then  $\hat{\mathbf{U}}^\circ = \hat{\mathbf{U}}_1 \hat{\mathbf{U}}_2$  and  $\hat{\mathbf{V}}^\circ = \hat{\mathbf{V}}_1 \hat{\mathbf{V}}_2$  satisfies eq. (C10).

Equation (C11) can be verified by direct multiplication. Inserting  $\mathbf{A} \hat{\mathbf{V}} = \hat{\mathbf{U}} \hat{\mathbf{\Sigma}} + \mathbf{E}_U$  leads to eq. (C12). Similarly we get eq. (C13) by inserting the  $\mathbf{A}^H \hat{\mathbf{U}} = \hat{\mathbf{V}} \hat{\mathbf{\Sigma}}^H + \mathbf{E}_V$ . ■

**Theorem C17: (NPSVD accuracy)** *Let the conditions be as in lemma C16, and assume that no element of  $\mathbf{M}_U$  or  $\mathbf{M}_V$  exceeds  $T$  in magnitude, where  $kT = \kappa < 1$ . Then there exists a permutation,  $P(i)$ , that maps the  $k$  numerical triplets,  $\hat{\sigma}_i$ , to exact triplets,  $\sigma_{P(i)}$ , of  $\mathbf{A}$  that satisfies:*

$$|\hat{\sigma}_i - \sigma_{P(i)}| \leq \hat{\sigma}_i \frac{2\kappa}{1-\kappa} + (\|\mathbf{E}_U\|_2 + \|\mathbf{E}_V\|_2) \frac{1+\kappa}{1-\kappa}$$

PROOF Let  $\check{\sigma}_i$  be the singular values of  $\mathbf{C} = \check{\mathbf{U}}^H \mathbf{A} \check{\mathbf{V}}$ . Using Ostrowski's theorem on eq. (C11) gives  $\check{\sigma}_i = \xi_1 \sigma_i$ , where  $1 - \kappa \leq |\xi_1| \leq 1 + \kappa$  by theorem C10.

Applying Ostrowski's theorem on the first term in eq. (C12) followed by Weyl's theorem gives  $\check{\sigma}_i = \xi_2 \hat{\sigma}_{P^{-1}(i)} + \xi_3$ , where  $P(i)$  is a permutation,  $1 - \kappa \leq \xi_2 \leq 1 + \kappa$ , and  $|\xi_3| \leq \|\mathbf{G}_U\|_2$ .

Eliminating  $\check{\sigma}_i$  from these expressions leads to:

$$\hat{\sigma}_{P^{-1}(i)} - \sigma_i = (1 - \xi_2/\xi_1) \hat{\sigma}_{P^{-1}(i)} - \xi_3/\xi_1$$

or, if we invert the perturbation,

$$(C14) \quad \hat{\sigma}_i - \sigma_{P(i)} = (1 - \xi_2/\xi_1) \hat{\sigma}_i - \xi_3/\xi_1$$

Inserting the extreme values for  $\xi_1$  and  $\xi_2$  shows that  $(1 - \kappa)/(1 + \kappa) \leq \xi_2/\xi_1 \leq (1 + \kappa)/(1 - \kappa)$ , which leads to  $|1 - \xi_2/\xi_1| \leq 2\kappa/(1 - \kappa)$ .

To bound  $|\xi_3/\xi_1|$  we use corollary C2 on  $\mathbf{G}_U$ :

$$|\xi_3/\xi_1| \leq \max\{\|\mathbf{E}_U\|_2, \|\mathbf{E}_V\|_2\} \frac{1}{1-\kappa} + \frac{1+\kappa}{1-\kappa} \|\mathbf{E}_U\|_2$$

An analogous treatment of eq. (C13) gives:

$$|\xi_3/\xi_1| \leq \max\{\|\mathbf{E}_U\|_2, \|\mathbf{E}_V\|_2\} \frac{1}{1-\kappa} + \frac{1+\kappa}{1-\kappa} \|\mathbf{E}_V\|_2$$

Choosing the lowest of these bounds:

$$\begin{aligned} |\xi_3/\xi_1| &\leq \max\{\|\mathbf{E}_U\|_2, \|\mathbf{E}_V\|_2\} \frac{1}{1-\kappa} + \min\{\|\mathbf{E}_U\|_2, \|\mathbf{E}_V\|_2\} \frac{1+\kappa}{1-\kappa} \\ &\leq (\|\mathbf{E}_U\|_2 + \|\mathbf{E}_V\|_2) \frac{1+\kappa}{1-\kappa} \end{aligned}$$

Inserting all these bound in eq. (C14) completes the proof. ■

Although theorem C17 may seem unwieldy, it bounds the difference between the exact singular values and those of an NPSVD in terms of quantities that are relatively easy to work with in a program.

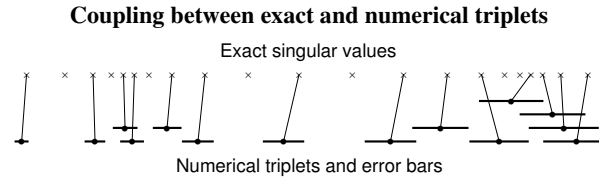
## C8 Triplet accuracy and aliasing

The singular triplets are meant to represent the matrix  $\mathbf{A}$ , so we want to interpret the singular triplets as approximations to exact triplets. This leads to the following requirements, illustrated in fig. C.1:

1. Each numerical triplet should map to an exact triplet of  $\mathbf{A}$ .
2. The difference between the singular value of a numerical triplet and the exact one it maps to should be bounded by a small multiple of  $\|\mathbf{A}\|_2 \epsilon_M$ .
3. The exact triplets in the map should be exactly orthogonal, i.e. the exact triplets should be unique.

**Figure C.1:**

All numerical triplets should map to a unique exact triplet.



Theorem C17 is the main vehicle we will use to prove that these requirements are met. However, by itself it leads to unnecessarily large bounds on the singular values. If we apply it to a single triplet instead, we get the following result.

**Corollary C18: (Single triplet accuracy)** Let  $(\hat{\mathbf{u}}_i, \hat{\mathbf{v}}_i, \hat{\sigma}_i)$  be a numerical triplet where  $\|\mathbf{A}\hat{\mathbf{v}}_i - \hat{\sigma}\hat{\mathbf{u}}_i\| \leq t_u$  and  $\|\mathbf{A}^H\hat{\mathbf{u}}_i - \hat{\sigma}\hat{\mathbf{v}}_i\| \leq t_v$ . Further assume that  $\hat{\mathbf{u}}_i$  and  $\hat{\mathbf{v}}_i$  are normalized, so that  $1 - \epsilon_o \leq \|\hat{\mathbf{u}}_i\|_2, \|\hat{\mathbf{v}}_i\|_2 \leq 1 + \epsilon_o$ . If  $\epsilon_o \leq 0.05$  then there exist an exact singular triplet of  $\mathbf{A}$ ,  $(\mathbf{u}_j, \mathbf{v}_j, \sigma_j)$ , so that

$$|\sigma_j - \hat{\sigma}_i| \leq 1.11(t_u + t_v) + 2.11|\hat{\sigma}_i|\epsilon_o$$

This asserts that the computed singular values are accurate as long as the numerical triplets maps to unique exact triplets. However, if we are given a set of numerical triplets and the intervals allowed by the corollary does not overlap, then the exact triplets *must* be unique.

If two singular triplets have overlapping ranges, they may both have a large component corresponding to the same exact triplet we wish to extract. We say that such triplets *alias* each other. In this case we must go back to theorem C17 and include all triplets that may potentially alias the same singular value.

## C9 One-sided reorthogonalization

One interesting observation is that theorem C15 and C17 are, in some sense, reciprocal. The first states that if either the right or left numerical vectors are kept orthogonal at level  $\epsilon_M$ , then the other vectors are roughly orthogonal at level  $T \approx \epsilon_M/\hat{\sigma}$ . On the other hand, theorem C17 says that to compute the singular values accurately (relative to the largest singular value), the level of orthogonality need only satisfy  $2kT\hat{\sigma} \approx \epsilon_M$ .

This leads us to think that keeping one set of triplets orthogonal suffices to compute singular values to full accuracy. The following theorem is a precise statement of this argument.

**Theorem C19: (One-sided orthogonality accuracy)** Assume a set of sorted numerical triplets,  $(\hat{\mathbf{u}}_i, \hat{\mathbf{v}}_i, \sigma_i)$ ,  $\hat{\sigma}_1 \geq \hat{\sigma}_2 \geq \dots \geq \hat{\sigma}_k$ , satisfies  $\|\mathbf{A}\hat{\mathbf{v}}_i - \hat{\sigma}_i\hat{\mathbf{u}}_i\| \leq t_u$  and  $\|\mathbf{A}^H\hat{\mathbf{u}}_i - \hat{\sigma}_i\hat{\mathbf{v}}_i\| \leq t_v$ . Furthermore, assume that all pairs satisfies either  $|\hat{\mathbf{u}}_j^H\hat{\mathbf{u}}_i| \leq T$  or  $|\hat{\mathbf{v}}_j^H\hat{\mathbf{v}}_i| \leq T$ . Finally assume that  $\kappa_k = kT + k(t_u + t_v)/\hat{\sigma}_k < 1$ .

Then the numerical triplets correspond to unique exact triplets of  $\mathbf{A}$ ,  $\hat{\sigma}_i \approx \sigma_{P_k(i)}$ , with the difference bounded by:

$$|\hat{\sigma}_i - \sigma_{P_k(i)}| \leq \hat{\sigma}_i \frac{2kT}{1 - \kappa_k} + 2k \left( \frac{\hat{\sigma}_i}{\hat{\sigma}_k} \right) (t_u + t_v) + (\|\mathbf{E}_U\|_2 + \|\mathbf{E}_V\|_2) \frac{1 + \kappa_k}{1 - \kappa_k}$$

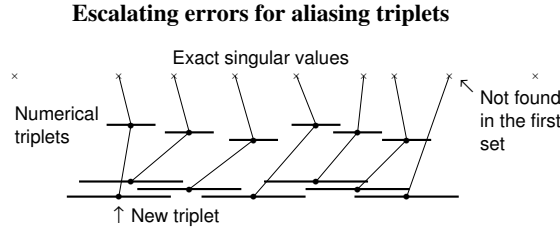
PROOF

In the case where  $|\mathbf{M}_U| \leq T$  we have, from theorem C15, that  $|\mathbf{M}_V| \leq T + (t_u + t_v)/\min\{\hat{\sigma}_i, \hat{\sigma}_j\}$ . Otherwise we have  $|\mathbf{M}_V| \leq T$ , and we similarly get  $|\mathbf{M}_U| \leq T + (t_u + t_v)/\min\{\hat{\sigma}_i, \hat{\sigma}_j\}$ . In either case theorem C17 applies with  $\kappa = \kappa_k$ , which is the stated result. ■

The analysis would be complete had it not been for the term  $\hat{\sigma}_i/\hat{\sigma}_k$ . We can eliminate this by setting  $k = i$ , but there is no guarantee that  $P_k(i)$  is the same for all  $k \geq i$ .

The problem is illustrated in fig. C.2. At some point we have a number of triplets determined accurately, but when we introduce a new triplet it may not be possible to map the previous triplets without increasing their error-bars substantially. In the example the escalation occurs even though the triplets are added in order of strictly decreasing singular values. Also, the new triplet can be arbitrarily close to an exact singular value.

**Figure C.2:** Adding a numerical triplet may cause the errors of previously triplets to increase.



The cure is to give up the pure one-sided approach and orthogonalize each triplet against all potentially aliasing triplets. This enforces the condition that  $\kappa_i \approx \kappa_k$  (by lowering  $\kappa_k$ ) if the error in the  $k$ 'th triplet could escalate to triplet  $i$ . With this modification the factor  $\hat{\sigma}_i/\hat{\sigma}_k$  vanishes.

The key observation is that we only need to keep both left and right singular vectors for a few triplets at a time. This means that the singular values can be computed to full machine precision with memory that scales in the smallest dimension of  $\mathbf{A}$ .



# BBTools reference

*This appendix contains a list of the functions provided with BBTools v0.2.4b, released 16-Sep-2005. Please note that function provided for compatibility is not addressed here. See chapter 10 for compatibility with MATLAB®.*

## List of categories and tables

Getting started with BBTools . . . . .	D.1
Elementary black-box operators . . . . .	D.2
Toeplitz and related operators . . . . .	D.3
Image- and hypercube operators . . . . .	D.4
Tomographic operators . . . . .	D.5
Miscellaneous operators . . . . .	D.6
Black-box creation, manipulation, and destruction . . . . .	D.7
Black-box information and verification . . . . .	D.8
Tests and probes for iterative algorithms . . . . .	D.9
Explicit black-box versions of standard functions . . . . .	D.10
Statistics . . . . .	D.11
Miscellaneous support functions . . . . .	D.12
Computing with black-box operators . . . . .	D.13
Equation solving and regularization . . . . .	D.14
SVD computations . . . . .	D.15

**Table D.1:**  
*Getting started  
with BBTools.*

Function	Description
bbdoc	Start online documentation for the toolbox
bbinstall	(GUI) Install, uninstall, download, and upgrade

**Table D.2:**  
*Elementary black-  
box operators.*

Function	Description
bbmatrix	Create black-box from a Matlab matrix
bbinv	Create black-box to invert a Matlab matrix
bbindex	Create indexing black-box
bbeye	Create identity black-box
bbdiag	Create diagonal black-box
bbconst	Create black-box with constant elements
bbzeros	Create black-box of zeros
bbmean	Create averaging black-box

**Table D.3:**  
*Toeplitz and related  
operators.*

Function	Description
bbtoeplitz	Create Toeplitz black-box
bbdbltoeplitz	Create doubly Toeplitz black-box
bbblktoepls	Create sparse block-Toeplitz type-1 black-box
bbhankel	Create Hankel black-box

**Table D.4:**  
*Image- and hyper-  
cube operators.*

Function	Description
bbflipdim	Create black-box that flips a specified dimension
bbpermute	Create dimension-permuting black-box
bbredim	Create redimensioning black-box
bbrescale	Create rescaling black-box
bbconvn	Create black-box for $N$ -dimensional convolution

**Table D.5:**  
*Tomographic oper-  
ators.*

Function	Description
bbtomoemit	Create black-box model of an emission tomograph
bbbradon	Create black-box for a Radon transform
bbfbp	Create inverse of Radon using filtered backprojection

**Table D.6:**  
*Miscellaneous op-  
erators.*

Function	Description
bbhadamard	Create Hadamard black-box
bbaplacian	Create a finite-difference Laplace operator

**Table D.7:**  
*Black-box cre-  
ation, manipula-  
tion, and destruc-  
tion.*

Function	Description
blackbox	Create black-box operator
bbclear	Remove a black-box without leaving garbage
bbqualify	Add informational qualifier to a black-box
bbimplify	Simplify a black-box operator
bbinline	Inline and freeze a black-box operator
get	Get black-box parameter
set	Set black-box parameter
isa	Check type of matrix operator

	<b>Function</b>	<b>Description</b>
<b>Table D.8:</b> <i>Black-box information and verification.</i>	bbinspect	(GUI) Utility to explore an evaluation tree
	bbshowtree	Show evaluation tree for a black-box
	bbtauest	Estimate accuracy of a black-box statistically
	bbcompare	Compare two operators for equality

	<b>Function</b>	<b>Description</b>
<b>Table D.9:</b> <i>Tests and probes for iterative algorithms.</i>	bbnoisy	Create a noisy version of an operator
	bbcount	Create black-box that counts invocations
	bbcountstat	Get count statistics
	bbcountreset	Reset counts statistics

	<b>Function</b>	<b>Description</b>
<b>Table D.10:</b> <i>Explicit black-box versions of standard functions.</i>	bbadd	Add a number of operators
	bbprod	Form a product of operators and scalars
	bbkron	Form Kronecker-product of two operators
	bbrepmat	Replicate and tile an operator

	<b>Function</b>	<b>Description</b>
<b>Table D.11:</b> <i>Statistics.</i>	bbrand	Draw random samples from a given distribution
	bbpca	Principal Component Analysis
	bbemipoisson	Solve linear system with Poisson noise

	<b>Function</b>	<b>Description</b>
<b>Table D.12:</b> <i>Miscellaneous support functions.</i>	bbkernel	Make a standard convolutional kernel
	bbphantom	Make a standard test-image
	bbimagein	Display intensity-image with gamma-correction

	<b>Function</b>	<b>Description</b>
<b>Table D.13:</b> <i>Computing with black-box operators.</i>	bbmult	Multiply black-box with a matrix
	bbmultq	Quick evaluation of product (no checks)
	bbmultqh	Quick evaluation of adjoint product (no checks)
	bbfunc	Get functions for matrix-vector products
	bbfuncm	Get functions for matrix-matrix products

	<b>Function</b>	<b>Description</b>
<b>Table D.14:</b> <i>Equation solving and regularization.</i>	bbpresolve	Prepare iterative solution of a linear system
	bbsolve	Compute a solution of a linear system
	bbregsolve	Compute a regularized solution of a linear system
	bbsolveopt	Create options for solving a linear system
	bbsolvemem	Estimate memory usage for solving a linear system
	bbllcurve	Compute L-curve and estimate regularization parameter

	<b>Function</b>	<b>Description</b>
<b>Table D.15:</b> <i>SVD computations.</i>	bbsvds	Compute singular values/triplets of an operator
	bbsvdf	Compute singular triplets of an operator with call-back
	bbsvdopt	Create options for computing an SVD
	bbsvdmem	Estimate memory usage for an SVD computation

# Bibliography

- [1] Helmut Alt, Bernd Behrends, and Johannes Blömer. Approximate matching of polygonal shapes. In *Proceedings of the 7th Annual Symposium on Computational Geometry*, pages 186–193. ACM Press, 1991. ISBN 0-89791-426-0.
- [2] Edward Anderson, Zhaojun Bai, Christian Heinrich Bischof, L. Susan Blackford, James Demmel, Jack J. Dongarra, Jeremy Du Croz, Anne Greenbaum, Sven Hammarling, Alan McKenney, and Danny Chris Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, PA, third edition, 1999. ISBN 0-89871-447-8.  
▶ <http://www.netlib.org/lapack/lug/index.html>.
- [3] Babak A. Ardekani, Michael Braun, Brian F. Hutton, Iwao Kanno, and Hidehiro Iida. Minimum cross-entropy reconstruction of PET images using prior anatomical information. *Physics in Medicine and Biology*, 41(11):2497–2517, 1996.  
▶ <http://www.iop.org/EJ/abstract/0031-9155/41/11/018>.
- [4] Richard Barrett, Michael W. Berry, Tony F. Chan, James Demmel, June M. Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Charles Romine, and Henk A. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. Web, 2nd edition, 1994. **Note:** The first edition is also published by SIAM, 1993, ISBN 0-89871-328-5.  
▶ [http://www.netlib.org/linalg/html\\_templates/Templates.html](http://www.netlib.org/linalg/html_templates/Templates.html).
- [5] Bernard Bendriem and David W. Townsend, editors. *The Theory and Practice of 3D PET*. Kluwer Academic Publishers, 1998. ISBN 0-7923-5108-8.
- [6] Michael W. Berry. Large-scale sparse singular value computations. *International Journal of Supercomputer Applications*, 6(1):13–49, 1992. ISSN 0890-2720.  
▶ <http://www.netlib.org/svdpack/ijsa.ps>.
- [7] Michael W. Berry. *SVDPACK*. University of Tennessee, 1992.  
▶ <http://www.netlib.org/svdpack/index.html>.
- [8] Christopher M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1995. ISBN 0-19-853864-2.
- [9] Åke Björck, Eric Grimme, and Paul Van Dooren. An implicit shift bidiagonalization algorithm for ill-posed systems. *BIT*, 34(4):510–534, 1994.
- [10] L. Susan Blackford, Jaeyoung Choi, Andrew J. Cleary, Eduardo Francisco D'Azevedo, James Demmel, Inderjit Singh Dhillon, Sven Hammarling, Greg Henry, Antoine Petitet, Ken Stanley, David William Walker, and Richard Clint Whaley. *ScaLAPACK Users' Guide*. SIAM, Philadelphia, PA, 1997. ISBN 0-89871-397-8.  
▶ <http://www.netlib.org/scalapack/slug/>.
- [11] James E. Bowsher, Valen E. Johnson, Timothy G. Turkington, Ronald J. Jaszczak, Carey E. Floyd, Jr, and Edward R. Coleman. Bayesian reconstruction and use of anatomical a priori information for emission tomography. *IEEE Transactions on Medical Imaging*, 15(8):673–686, 1996.
- [12] Jerrold T. Bushberg, J. Anthony Seibert, Edwin M. Jr. Leidholdt, and John M. Boone. *The Essential Physics of Medical Imaging*. Lippincott Williams & Wilkins, 2nd edition, 2002. ISBN 0-683-30118-7.
- [13] Daniela Calvetti, L. Reichel, and Danny Chris Sorensen. An implicitly restarted Lanczos method for large symmetric eigenvalue problems. *Electronic Transactions on Numerical Analysis*, 2:1–21, 1994.

- [14] Jon F. Claerbout. *Image Estimation by Example*. Web, 2004.  
▶ <http://sepwww.stanford.edu/sep/prof/toc.html/index.html>.
- [15] James Weldon Demmel. *Applied Numerical Linear Algebra*. SIAM, Philadelphia, PA, 1997. ISBN 0-89871-389-7.
- [16] Stephen E. Derenzo, William W. Moses, Ronald H. Huesman, and Thomas F. Budinger. Critical instrumentation for resolution smaller than 2mm, high sensitivity brain PET. In Kazuo Uemura, Terry Jones, Niels A. Lassen, and Iwao Kanno, editors, *Quantification of Brain Function*, pages 25–37. Elsevier Science Publishers B.V., 1993. ISBN 0-444-89859-x.
- [17] Jack J. Dongarra and David William Walker. The design of linear algebra libraries for high performance computers. Technical Report UT-CS-93-188, University of Tennessee, 1993.  
▶ <http://www.netlib.org/lapack/lawnspdf/lawn58.pdf>.
- [18] D. A. Dougherty, J. Nissanov, and Gene R. Gindi. Construction details of an autoradiographic-based rat brain phantom for emission tomography. In *Nuclear Science Symposium, 1998. Conference Record.*, volume 2, pages 1250–1254. IEEE, 1998. ISBN 0-7803-5021-9.
- [19] Robert Joseph English. *SPECT, Single-Photon Emission Computed Tomography: a Primer*. Society of Nuclear Medicine, 3rd edition, 1995. ISBN 0-932004-43-1.
- [20] Jeffrey A. Fessler. Penalized weighted least squares image reconstruction for PET. *IEEE Transactions on Medical Imaging*, 13(2):290–300, 1994.
- [21] Jeffrey A. Fessler. TERSE – a transmission and emission reconstruction environment for SPECT. Web, 1996.  
▶ <http://www.eecs.umich.edu/~fessler/terse/>.
- [22] Jeffrey A. Fessler. Statistical image reconstruction methods for transmission tomography. In Milan Sonka and J. Michael Fitzpatrick, editors, *Handbook of Medical Imaging: medical image processing and analysis*, volume 2, pages 1–70. The International Society for Optical Engineering, 2000. ISBN 0-8194-3622-4.
- [23] Jeffrey A. Fessler. *ASPIRE – A Sparse Iterative Reconstruction Library*. The University of Michigan, 2003.  
▶ <http://www.eecs.umich.edu/~fessler/aspire/index.html>.
- [24] Free Software Foundation, Inc. GNU general public license, version 2, 1991. In [68], pages 195–200.  
▶ <http://www.fsf.org/copyleft/gpl.html>.
- [25] Philip W. Goetz, editor. *Encyclopædia Britannica*. Encyclopædia Britannica, Inc., 15th edition, 1991. ISBN 0-85229-529-4.
- [26] Gene Howard Golub and Charles Francis Van Loan. *Matrix Computations*. John Hopkins University Press, 2nd edition, 1989. ISBN 0-8018-3739-1.
- [27] Gene Howard Golub and Charles Francis Van Loan. *Matrix Computations*. John Hopkins University Press, 3rd edition, 1996. ISBN 0-8018-5414-8.
- [28] S.F. Haber, Stephen E. Derenzo, and D. Uber. Application of mathematical removal of positron range blurring in positron emission tomography. *IEEE Transactions on Nuclear Science*, 27(3):1293–1299, 1990.  
▶ <http://breast.lbl.gov/~wwinstr/publications/Papers/range.pdf>.
- [29] Per Christian Hansen. Regularization tools: A matlab package for analysis and solution of discrete ill-posed problems. *Numerical Algorithms*, 6:1–35, 1994.
- [30] Per Christian Hansen. *Rank-Deficient and Discrete Ill-Posed Problems*. SIAM, 1998. ISBN 0-89871-403-6.

- [31] Per Christian Hansen. Regularization tools. Technical Report IMM-REP-98-6, Technical University of Denmark, 2001.  
▶ <http://www.imm.dtu.dk/~pch/Regutools/>.
- [32] Michiel E. Hochstenbach. A Jacobi–Davidson type SVD method. *SIAM Journal on Scientific Computing*, 23(2):606–628, 2001. ISSN 1064-8275.
- [33] Esben Høgh-Rasmussen. *BBTools – a Matlab Toolbox for Black-Box Computations*. Neurobiology Research Unit, Copenhagen University Hospital, 2005.  
▶ <http://nru.dk/software/bbtools/>.
- [34] Esben Høgh-Rasmussen. A cache-friendly kin to bidiagonalization with orthogonalization and restart, 2005. Submitted to BIT.
- [35] Søren Holm, Peter Aundal Toft, and Mikael Jensen. Estimation of the noise contributions from blank, transmission and emission scans in PET. In *IEEE Medical Imaging Conference*, pages 1470–1474, October 1995.
- [36] John H. Hubbell and Stephen M. Seltzer. Tables of x-ray mass attenuation coefficients and mass energy-absorption coefficients. Technical report, National Institute of Standards and Technology, 1997. Originally published as NISTIR 5632, NIST, Gaithersburg, MD (1995).  
▶ <http://physics.nist.gov/xaamdi>.
- [37] Harold Malcolm Hudson and Richard S. Larkin. Accelerated image reconstruction using ordered subsets of projection data. *IEEE Transactions on Medical Imaging*, 13(4):601–609, 1994.
- [38] Michael Jacobsen. *Modular Regularization Algorithms*. PhD thesis, 2004.  
▶ <http://www2.imm.dtu.dk/pubdb/p.php?id=3380>.
- [39] Avinash C. Kak and Malcolm Slaney. *Principles of Computerized Tomographic Imaging*. IEEE Press, 1988. ISBN 0-8794-2198-3.  
▶ <http://www.slaney.org/pct/pct-toc.html>.
- [40] Satoshi Kawata and Orhan Nalcioglu. Constrained iterative reconstruction by the conjugate gradient method. *IEEE Transactions on Medical Imaging*, 4(2):65–71, 1985.
- [41] John Kirman. *Good Style, Writing for Science and Technology*. E & FN Spon, 1994. ISBN 0419171908.
- [42] Donald Ervin Knuth. *Literate Programming*. Center for the Study of Language and Inf., 1992. ISBN 0937073814.
- [43] Samuel Kotz, Normal Lloyd Johnson, and Campbell B. Read, editors. *Encyclopedia of Statistical Sciences*, volume 8. John Wiley & Sons, Inc., 1988. ISBN 0-471-05556-5.
- [44] Erwin Kreyszig. *Advanced Engineering Mathematics*. John Wiley & Sons, Inc., 7th edition, 1993. ISBN 0-47-155380-8.
- [45] Kenneth Lange and Richard Carson. EM reconstruction algorithms for emission and transmission tomography. *Journal of Computer Assisted Tomography*, 8(2):306–316, 1982.
- [46] Rasmus Munk Larsen. Lanczos bidiagonalization with partial reorthogonalization. Technical Report DAIMI PB-357, Department of Computer Science, Aarhus University, 1998.  
▶ <http://sun.stanford.edu/~rmunk/PROPACK/paper.ps.gz>.
- [47] Rasmus Munk Larsen. *PROPACK*. Stanford University, 2005.  
▶ <http://sun.stanford.edu/~rmunk/PROPACK/>.
- [48] Soo-Jin Lee, Anand Rangarajan, and Gene Gindi. Bayesian image reconstruction in SPECT using higher order mechanical models as priors. *IEEE Transactions on Medical Imaging*, 14(4):669–680, 1995.

- [49] Richard Bruno Lehoucq and Jennifer A. Scott. An evaluation of Arnoldi based software for sparse non-symmetric eigenproblems. Technical Report RAL-TR-96-023, Rutherford Appleton Laboratory, Didcot, England, 1996.
- [50] Richard Bruno Lehoucq, Danny Chris Sorensen, and Chao Yang. *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. Rice University, 1997.  
▶ <http://www.caam.rice.edu/software/ARPACK/>.
- [51] Ren-Cang Li. Relative perturbation theory: I. eigenvalue and singular value variations. *SIAM Journal on Matrix Analysis and Applications*, 19(4):956–982, 1998.
- [52] Kanti V. Mardia, John T. Kent, and John M. Bibby. *Multivariate Analysis*. Academic Press, London, 1995. ISBN 0-12-471252-5.
- [53] Ronald B. Morgan. GMRES with deflated restarting. *SIAM Journal on Scientific Computing*, 24(1): 20–37, 2002.
- [54] Alan Victor Oppenheim and Ronald W. Schaffer. *Discrete-Time Signal Processing*. Prentice Hall, Inc., 1989. ISBN 0-13-216771-9.
- [55] Christopher Conway Paige and Michael Alan Saunders. LSQR: an algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software (TOMS)*, 8(1):43–71, 1982. ISSN 0098-3500.
- [56] Marko Petkovsek, Herbert Wilf, and Doron Zeilberger. *A = B*. AK Peters, Ltd., 1996. ISBN 1-56881-063-6.  
▶ <http://www.cis.upenn.edu/~wilf/AeqB.html>.
- [57] Peter Alshede Philipsen, Lars Kai Hansen, and Peter Aundal Toft. Mean field reconstruction with snaky edge hints. In *Proceedings of the Interdisciplinary Inversion Conference on Methodology, Computation and Integrated Applications IIC'95 1996*, pages 312–319. Springer Verlag, Berlin, 1996.
- [58] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical recipes in C: The art of scientific computing*. Cambridge University Press, New York, 2nd edition, 1992. ISBN 0-521-43108-5.
- [59] Johann Karl August Radon. Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten. *Berichte Sachsische Akademie der Wissenschaften*, 69:262–267, 1917.  
**Note:** The title roughly translates to: “On the determination of functions from the integrals on certain manifolds”.
- [60] Sheldon M. Ross. *Introduction to probability and statistics for engineers and scientists*. John Wiley & Sons, Inc., 1987. ISBN 0-471-81752-X.
- [61] Yousef Saad and Henk A. van der Vorst. Iterative solution of linear systems in the 20th century. *Journal of Computational and Applied Mathematics*, 123(1–2):1–33, 2000.  
▶ <http://www.sciencedirect.com/science/article/B6TYH-41H9NjX-2/2/235bc014228315f21059108e5cef9940>.
- [62] Semiconductor Industry Association. *The International Technology Roadmap for Semiconductors*. ITRS, 2003 edition edition, 2003.  
▶ <http://public.itrs.net/>.
- [63] Semiconductor Industry Association. *The International Technology Roadmap for Semiconductors*. ITRS, 2004 update edition, 2004.  
▶ <http://public.itrs.net/>.
- [64] Lawrence A. Shepp and Yehuda Vardi. Maximum likelihood reconstruction for emission tomography. *IEEE Transactions on Medical Imaging*, 1:113–122, 1982.

- [65] Donald Lee Snyder and Michael I. Miller. The use of sieves to stabilise images produced with the EM algorithm for emission tomography. *IEEE Transactions on Nuclear Science*, 32(5):3864–3872, 1985.
- [66] Danny Chris Sorensen. Implicit application of polynomial filters in a k-step Arnoldi method. *SIAM Journal on Matrix Analysis and Applications*, 13(1):357–385, 1992.
- [67] Danny Chris Sorensen. Implicitly restarted Arnoldi/Lanczos methods for large scale eigenvalue calculations. Technical Report TR-96-40, Rice University, 1996.  
▶ <http://techreports.larc.nasa.gov/icase/1996/icase-1996-40.pdf>.
- [68] Richard M. Stallman. *Free Software, Free Society: Selected Essays of Richard M. Stallman*. Free Software Foundation, Inc., 2002. ISBN 1-882114-98-1.  
▶ <http://www.gnu.org/doc/book13.html>.
- [69] Gilbert Wright Stewart. *Matrix Algorithms: Basic decompositions*. SIAM, 1998. ISBN 0-89871-414-1.
- [70] Gilbert Wright Stewart and Ji-guang Sun. *Matrix Perturbation Theory*. Academic Press, 1990. ISBN 0-12-670230-6.
- [71] Gilbert Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 2nd edition, 1998. ISBN 0-9614088-5-5.
- [72] Peter Aundal Toft. *The Radon Transform - Theory and Implementation*. PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, 1996.  
▶ <http://pto.linux.dk/PhD/>.
- [73] Lloyd Nicholas Trefethen and David Bau III. *Numerical Linear Algebra*. SIAM, Philadelphia, PA, 1997. ISBN 0-89871-487-7.
- [74] Timothy G. Turkington. Introduction to PET instrumentation. *Journal of Nuclear Medicine Technology*, 29(1):4–11, 2001.
- [75] *SimSET – Simulation System for Emission Tomography*. University of Washington, 2004.  
▶ [http://depts.washington.edu/~simset/html/simset\\_home.html](http://depts.washington.edu/~simset/html/simset_home.html).
- [76] Eugene Veklerov and Jorge Llacer. The feasibility of images reconstructed with the method of sieves. *IEEE Transactions on Nuclear Science*, 37(2):835–841, 1990.
- [77] Gustav Konrad von Schulthess, editor. *Clinical Positron Emission Tomography, Correlation with Morphological Cross-Sectional Imaging*. Lipincott Williams & Wilkins, 2000. ISBN 0-7817-1756-6.
- [78] Steve Webb, editor. *The Physics of Medical Imaging*. Institute of Physics Publishing, 1988. ISBN 0-85274-349-1.
- [79] Wolfgang Weinrich and Hans-Joachim Kretschmann. *Cranial neuroimaging and clinical neuroanatomy: computed tomography and magnetic resonance imaging*. Thieme Medical Publishers, Inc., 1992.
- [80] Richard Clint Whaley, Antoine Petitot, and Jack J. Dongarra. Automated empirical optimization of software and the ATLAS project. *Parallel Computing*, 27(1–2):3–35, 2001.  
▶ <http://www.netlib.org/lapack/lawns/lawn147.ps>.
- [81] Peter Kjær Willendrup. Point-spread functions in tomography using filtered back-projection reconstruction. Master's thesis, University of Copenhagen, 2000.  
▶ <http://www.willendrup.org/thesis.pdf>.
- [82] Keith John Worsley, A.C. Evans, S. Marrett, and P. Neelin. A three-dimensional statistical analysis for cbf activation studies in human brain. *Journal of Cerebral Blood Flow and Metabolism*, 12:900–918, 1992.  
▶ <http://www.math.mcgill.ca/~keith/jcbf/jcbf.abstract.html>.



- [83] Keith John Worsley, S. Marrett, P. Neelin, A.C. Vandal, K.J. Friston, and A.C. Evans. A unified statistical approach for determining significant signals in images of cerebral activation. *Human Brain Mapping*, 4: 58–73, 1996.  
▶ <http://www.math.mcgill.ca/~keith/unified/unified.abstract.html>.
- [84] Mehmet Yavuz. Statistical tomographic image reconstruction methods for randoms-precorrected PET measurements. Technical report, The University of Michigan, 2000. Ph.D.thesis.
- [85] Mehmet Yavuz and Jeffrey A. Fessler. Objective functions for tomographic reconstruction from randoms-precorrected PET scans. In *Proceedings of IEEE Nuclear Science Symposium and Medical Imaging Conference*, volume 2, pages 1067–71. IEEE, 1996.
- [86] Daniel Zwillinger, editor. *Standard Mathematical Tables and Formulae*. CRC Press, 30th edition, 1996. ISBN 0-8493-2479-3.

# Index

- 2D acquisition, 19
- 3D acquisition, 19
  
- absolute norm, 147
- accuracy
  - asymmetric, 99
- acknowledgments, v
- acquisition
  - continuous, 23
  - matrix, 127
  - step-and-shoot, 23
- acquisitions mode, 19
- active contour model, 56
- adaptive snake, 56
- algorithm
  - arnoldi, 88
  - back-project, 37
  - bidiag, 87
  - init, 85
  - iter, 146
  - randsolve, 108
  - reorthogonalize, 91
  - restart, 93
  - tridiag, 87
  - unit, 85
  - update, 85
- aliasing, 40, 101, 155
- Alzheimer's disease, iv
- anatomy, 58
- Arnoldi iteration, 87–88
- ARPACK, 93
- ART, 46
  - see also reconstruction.*
- artifact
  - star, 39, 41
  - streaks, 41
- artifacts, 52
- ASPIRE, 32
- asymmetric accuracy, 99
- attenuation, 12, 16
  - computing, 29
  - linear attenuation coefficient, 13
  - mass attenuation coefficient, 13
- autoradiography, 56
  
- back-projection, 35–37
- backwards compatibility, 53, 128
- basis, 30
- Bayesian inference, 56
  
- BBTools, 4, 6, 32, 37, 99, 113–123, 157–160
  - compatibility, 120
- beam divergence, 16, 26
- bidiag, 87 (*see algorithm*)
- bidiagonalization, 87 (*see Lanczos*), 97, 108
- bin, 23, 28, 30
- black-box operator, 32
- BLAS, 96
- BLAS levels, 96
- blocking, 96
  
- CAT, 9
  - see also CT.*
- Cauchy inequality, 149
- CGNR, 109
- Chebyshev polynomials, 91
- collimator, 14, 24
  - cone-beam, 15
  - fan-beam, 15
  - parallel hole, 14
  - pin-hole, 15, 26
  - sensitivity, 15
- collinearity, 17
- compact support, 22, 23, 55
- condition number
  - near-orthogonal matrix, 151
- consistent norm, 149
- continuous
  - acquisition, 23 (*see acquisition*)
- convergence
  - EM algorithm, 44
- Cormack, Allan MacLeod, 9
- coronal plane, 5
- coupling matrix, 80 (*see NPSVD*)
- coverage, 52
- CT, 12 (*see scanner*)
  
- dead-time, 18
- deflation
  - explicit, 102
  - implicit, 104
  - one-sided, 103
  - two-sided, 102
- discrepancy principle, 68
- discrete Picard condition, 111
- distribution
  - Gaussian, 33

- multinomial, 31
  - Poisson, 30
  - shifted Poisson, 32, 33
- divergence, 16 (*see beam divergence*)
- early stopping, 45
- EM, 42–46
  - Kullback-Leibler distance, 55
  - sieve, 46
- energy window, 16
- epilepsy, iv
- EU Fifth Framework Programme, v
- exact
  - shift strategy, 91
  - singular triplet, 79
- explicit
  - orthogonalization, 151
  - see also orthogonalization.*
  - restart, 109
- FBP, 38–42
- feasibility shell, 68
- Felix Hausdorff, 69
- Fessler, Jeffrey A., 32
- field of view, 23
- filter-factors, 59, 110
- filtered back-projection, 38 (*see FBP*)
- Fourier transform, 39
- FOV, 23 (*see field of view*), 36
- full reorthogonalization, 86
- full-width at half maximum, 10
- funding, v
- FWHM, 10
- Galerkin, 109
- gallery, 10
- Gauss-Seidel, 46
- Gaussian likelihood, 66
- Gaussian random fields, 54
- general public license, 6
- geometric correction, 28, 31
- GMRES, 108
- GNU, 6
- GPL, 6
- H:S (Hovedstadens Sygehusfællesskab),
  - v
- Hausdorff
  - distance, 69
  - Felix, 69
- homogeneity, 52, 57
- Hounsfield, Sir Godfrey Newbold, 9
- Hudson, Harold Malcolm, 43
- hybrid solver, 111
- hypercube, 30, 118
- Høgh-Rasmussen, Nis, v
- implicit
  - deflation, 104
  - orthogonality, 151
  - see also singular triplet.*
  - restarting, 98, 109
- implicit restart, 91
- initialism, 12
- isotropic, 52
- iterative
  - reconstruction, 43
  - refinement, 107
  - solver, 107
- Johann Karl August Radon, 22
- John Von Neumann, 148
  - see also Von Neumann.*
- Jørgensens og Gunnar Hansens Fond, v
- Kaczmarz method, 46
- Keith John Worsley, 54
- Kilmer, Misha, v
- Kronecker product, 40
- Krylov, 4, 86
  - see also Lanczos and Arnoldi.*
  - matrix, 86
  - restart, 90
  - subspace, 86
- Kullback-Leibler distance, 55
- Ky Fan's dominance theorem, 148
- Lanczos, 97, 108
  - bidiagonalization, 87
  - restart, 90
  - tridiagonalization, 86–87
  - unsymmetric tridiag., 88
- LAPACK, 96
- Laplace operator, 60
- Larkin, Richard S., 43
- latent semantic indexing, 104
- least-squares
  - matrix approximation, 150
  - model-weighted, 67
  - weighted, 67
- left residual factorization, 97, 108
- legat
  - Jeppe J. og hustru Ovita J., v
  - Jørgensens og Gunnar H., v
  - Otto Mønsted, v
- Leja points, 91, 93
- level 1–3, 96
- level of orthogonality, 150
- license
  - GPL, 6
- line of response, 22
- literate programming, 6

- LOR, 16, 22, 35  
 LSQR, 93, 112
- machine epsilon, 7  
 machine precision, 7
- matrix  
   least-squares approximation, 150  
   low-rank approximation, 150  
   *see also* *Schmidt-Mirsky*.  
   near-orthogonal, 150  
   small, 6  
   sparse, 98
- matrix structure, 87  
 MCI, iv (*see mild cognitive impair.*)  
 mean-field approximation, 56  
 median plane, 5  
 Mehmet Yavuz, 33  
 memory hierarchy, 96  
 method of sieves, 46  
 mild cognitive impairment, iv  
 MINRES, 108  
 Misha Kilmer, v  
 MLS, 67  
 model-weighted least-squares, 67  
 modularization, 97  
 mollifier methods, 58  
 Monte-Carlo simulations, 32  
 M $\mathcal{O}$ RE Tools, 123  
 Morgan, Ronald B., 97  
 $\mu$ -map, 14  
 multi-head, 14  
 multinomial distribution, 31
- near-orthogonal matrix, 150  
 Nis Høgh-Rasmussen, v  
 non-collinearity, 17  
 non-negative, 36  
 norm  
   absolute, 147  
   consistent, 149  
   Ky Fan, 148  
   max-norm, 149  
   padded matrix dominance, 148  
   unitarily invariant, 147
- normal equations, 108  
 normal form, 22  
 NPSVD, 4, 80  
   *see also* *SVD*, *Krylov*, *Arnoldi*,  
   and *Lanczos*.  
   accuracy, 153  
   coupling matrix, 80  
   diagonalization, 82  
   orthogonalize, 90  
   restart, 92  
   search vectors, 83  
   singular error matrix, 80  
   singular values, 153  
   terminology, 80  
   update, 82
- null-space, 44  
 numerical  
   notation, 7  
   partial SVD (NPSVD), 80  
   singular triplet, 79
- on-line subtraction, 18, 32  
 one-sided  
   deflation, 103  
   reorthogonalization, 155  
   *see also* *orthogonalization*.
- ordered subset, 43  
 orthogonal, 5  
   matrix, 150  
 orthogonal projector, 62  
 orthogonality  
   implicit, 151  
   *see also* *singular triplet*.  
   level, 143, 150  
   loss, 143  
 orthogonalization  
   explicit, 151  
   Gram-Schmidt, 88  
   one-sided, 155
- orthonormal, 5  
 OSEM, 43  
 Ostrowski's theorem, 147  
 Otto Mønsted's Fond, v
- partial SVD, 4, 79  
   *see also* *SVD*.  
 penalty function, 60  
 perturbation theory, 147  
 PET, 16 (*see scanner*)  
   model, 27  
 phantom, 21  
 photo multiplier tube, 18  
 Picard condition, 111  
 pixel, 30  
 pixelation, 45, 55  
 PMT, 18 (*see photo multiplier tube*)  
 point-spread function, 10, 129  
 Poisson  
   distribution, 30  
   Simeon Denis, 30  
 positron range, 17, 29  
 post-filter, 45, 58  
 post-solve, 111  
 pre-solve, 111  
 preconditioner, 41, 58, 83, 99  
 prior, 55

- programming
  - literate, 6
- projector, 62
- propaedeutic (*προπαιδευτική*), 3
- pseudo-inverse, 39
- PSF, 10, 52, 129 (*see point-spread function*)
- PSVD, 79, 79
  - see also SVD and NPSVD.*
  
- quantitative, 52
- quasi-linear reconstruction, 58
  
- Radon
  - discrete transform, 23
  - Johann Karl August, 22
  - transform, 22
- ramp filter, 39
- Random fields
  - Gaussian, 54
- randoms, 18
- reconstruction, 3
  - ART, 46
  - EM, 42–46
  - iterative, 43
- regularization, 58
- reorthogonalization, 85, 151
  - see also orthogonalization.*
- residual, 107
- residual factorization, 97
- resolution, 10, 57
- restart
  - exact shift strategy, 91
  - explicit, 109
  - implicit, 91, 98, 109
  - Lanczos, 90
  - NPSVD, 92
- ridge regression, 60
- Riemann-Lebesgue, 55
- ringing, 42
- Ronald B. Morgan, 97
- rounding residual, 97, 100, 108
  
- sagittal plane, 5
- sampling theorem, 55
- Savværksejer Jeppe Juhl og hustru Ovita
  - Juhls Mindelegat, v
- scan
  - transmission, 12
- scanner
  - abbreviations, 9
  - acquisitions mode, 19
  - CT, 12–14
  - gallery, 10
  - PET, 16–19
  - resolution, 10
  - septa, 19
  - SPECT, 14–16
    - multi-head, 14
  - tracer, 14
- Schmidt-Mirsky's theorem, 150
- Schwartz, 149 (*see Cauchy inequality*)
- scintillation crystal, 16
- search vectors, 83
- selective reorthogonalization, 85
- semi-orthogonal vectors, 86
- semi-orthogonality, 90
- septa, 19
- shifted Poisson distribution, 32
- sieve, 46
- SimSET, 32
- simulated annealing, 56
- singular triplet, 79, 151
  - accuracy, 155
  - exact, 79
  - implicit orthogonality, 151
  - numerical, 79
  - orthogonal, 79
  - orthogonality, 151, 152
- singular value decomposition (SVD), 79
- singular values
  - near-orthogonal matrix, 150
- SIR, 55
- SIRA, 55
- slant-stacking, 32
- small, 6
- solver, iterative, 107
- sparse matrix, 98
- spatial invariant, 52
- SPECT, 14 (*see scanner*)
  - model, 24
- star artifact, 39, 41 (*see artifact*)
- step-and-shoot, 23 (*see acquisition*)
- Stieltjes integrals, 40
- stopping criteria, 143
- streak artifact, 39, 41 (*see artifact*)
- surrogate function, 55
- SVD, 4, 79–105
  - NPSVD, 80
  - partial (PSVD), 79
  - perturbation, 147
  - singular triplet, 79
  - thin, 59
- system matrix, 30, 32
  
- theorem
  - Central Slice, 38
  - cond. of near-ortho. matrix, 151
  - EM convergence, 44
  - Fourier Slice, 38

- Ky Fan dominance, 148
- matrix approximation, 150
- NPSVD accuracy, 154
- Ostrowski, 147
- padded matrix dominance, 148
- projection dominance, 148
- sampling, 55
- Schmidt-Mirsky, 150
- triplet accuracy, 155
- triplet orthogonality, 152
- Von Neumann, 148
- Weyl, 147
- thin SVD, 59
- Tikhonov regularization, 59
- Toft, Peter Aundal, 32
- tomography, 3
- tracer
  - dose, 30
  - SPECT, 14
- transaxial plane, 5
- transform
  - Fourier, 39
  - Radon, 22
- transmission scan, 12
- transverse plane, 5
- tridiag, 87 (*see algorithm*)
- tridiagonalization, 86 (*see Lanczos*)
- triplet, 79 (*see singular triplet*)
- triplet aliasing, 101, 155
- two-sided deflation, 102
  
- unitarily invariant norm, 147
  - normalized, 148
  
- variability, 51
- Von Neumann
  - unit. invariant norm theorem, 148
  
- weighted least-squares, 60, 67
- Weyl's theorem, 147
- WLS, 67 (*see weighted least-squares*)
- Woodbury's formula, 108
- Worsley, Keith John, 54
  
- Yavuz, Mehmet, 33

# Nomenclature

## General notation

$x^*$	Complex conjugate of $x$ .
$\triangleq$	Definition, e.g. $x \triangleq 2$ means that $x = 2$ by definition.
$\epsilon_M$	Machine epsilon, the smallest number such that $1 + \epsilon_M$ rounds to 1.

## Matrix notation

All vectors are column vectors. Row vectors are described as transposed (conjugated) column vectors. Matrices are written in uppercase boldface letters, like  $\mathbf{A}$ , while vectors are written in lowercase, like  $\mathbf{x}$ .

Submatrices are indicated using MATLAB<sup>®</sup> notation. Thus the matrix  $\mathbf{A}_{(:,1:3)}$  consists of the first three columns of  $\mathbf{A}$ .

$\mathbf{A}^T$	Transpose.
$\mathbf{A}^H$	“Hermitian”, i.e. transpose-conjugated. Thus $\mathbf{A}^H = (\mathbf{A}^T)^*$ .
$\mathbf{A}^+$	The pseudo-inverse, or Moore-Penrose inverse, of the matrix $\mathbf{A}$ .
$\ \mathbf{A}\ $	Unspecified norm of $\mathbf{A}$ (see p. 149 for other norms).
$\mathbf{I}$	Identity matrix. When used in an expression, it automatically have the necessary dimensions.
$\mathbf{0}$	The zero-vector or -matrix of appropriate dimension.
$\text{range}(\mathbf{A})$	The space spanned by the columns of $\mathbf{A}$ : $\text{range}(\mathbf{A}) = \{\mathbf{y} : \mathbf{y} = \mathbf{A}\mathbf{x}\}$ , where $\mathbf{x}$ is taken over all possible vectors.
$\text{null}(\mathbf{A})$	Null-space of $\mathbf{A}$ : $\text{null}(\mathbf{A}) = \{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{0}\}$ .
$\text{diag}(\mathbf{x})$	Diagonal, i.e. $\mathbf{A} = \text{diag}(\mathbf{x})$ implies $\mathbf{A}_{(i,i)} = \mathbf{x}(i)$ and $\mathbf{A}_{(i,j)} = 0$ for $i \neq j$ .

## Sets

$\mathbb{R}$	Real numbers.
$\mathbb{C}$	Complex numbers.
$\mathbb{Q}$	Rational numbers.
$\mathbb{N}$	Natural numbers (non-negative integers).
$\mathbb{Z}$	Integers.

**Signal processing** $\mathcal{F}$ 

The Fourier-transform. If  $F(\omega)$  and  $f(x)$  are Fourier-pairs, then we have:

$$F(\omega) = \mathcal{F}\{f(x)\} = \int_{-\infty}^{\infty} f(x)e^{-j\omega x} dx$$

$$f(x) = \mathcal{F}^{-1}\{F(\omega)\} = \int_{-\infty}^{\infty} F(\omega)e^{+j\omega x} d\omega / (2\pi)$$

**Statistics** $\mathcal{E}\{\}$ 

Expectation. E.g.  $\mathcal{E}\{x\}$  is the mean value of the stochastic variable  $x$ .

var

Variance.  $\text{var}\{x\} = \mathcal{E}\{(x - \mathcal{E}\{x\})^2\}$ .

**Miscellaneous** $\mathcal{O}$ 

Asymptotic upper bound. The function  $g(x)$  is in the set  $\mathcal{O}(f(x))$  if there are positive constants  $c$  and  $x_0$  such that  $0 \leq f(x) \leq cg(x)$  for all  $x > x_0$ .

If an algorithm runs in  $\mathcal{O}(n^2)$  time, it means that the time taken for the algorithm is less than  $cn^2$  for some constant  $c$ , when  $n$  is sufficiently large.