Numerical construction of Green's functions in high dimensional elliptic problems with variable coefficients and analysis of renewable energy data via sparse and separable approximations

David Joseph Biagioni
University of Colorado at Boulder, biagioni@colorado.edu

Follow this and additional works at: https://scholar.colorado.edu/appm_gradetds

Part of the Applied Mathematics Commons

Recommended Citation
https://scholar.colorado.edu/appm_gradetds/29
Numerical construction of Green’s functions in high dimensional elliptic problems with variable coefficients and analysis of renewable energy data via sparse and separable approximations

by

David Joseph Biagioni

B.S. Georgia Institute of Technology, 2003

M.S. University of Colorado at Boulder, 2011

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Applied Mathematics

2012
This thesis entitled:
Numerical construction of Green’s functions in high dimensional elliptic problems with variable coefficients
and analysis of renewable energy data via sparse and separable approximations
written by David Joseph Biagioni
has been approved for the Department of Applied Mathematics

______________________________
Gregory Beylkin

______________________________
Alireza Doostan

Date __________________________

The final copy of this thesis has been examined by the signatories, and we find that both the content and
the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.
This thesis consists of two parts. In Part I, we describe an algorithm for approximating the Green's function for elliptic problems with variable coefficients in arbitrary dimension. The basis for our approach is the separated representation, which appears as a way of approximating functions of many variables by sums of products of univariate functions. While the differential operator we wish to invert is typically ill-conditioned, its conditioning may be improved by first applying the Green's function for the constant coefficient problem. This function may be computed either numerically or, in some case, analytically in a separated format. The variable coefficient Green's function is then computed using a quadratically convergent iteration on the preconditioned operator, with sparsity maintained via representation in a wavelet basis. Of particular interest is that the method scales linearly in the number of dimensions, a feature that very desirable in high dimensional problems in which the curse of dimensionality must be reckoned with.

As a corollary to this work, we described a randomized algorithm for maintaining low separation rank of the functions used in the construction of the Green's function. For certain functions of practical interest, one can avoid the cost of using standard methods such as alternating least squares (ALS) to reduce the separation rank. Instead, terms from the separated representation may be selected using a randomized approach based on matrix skeletonization and the interpolative decomposition. The use of random projections can greatly reduce the cost of rank reduction, as well as calculation of the Frobenius norm and term-wise Gram matrices.

In Part II of the thesis, we highlight three practical applications of sparse and separable approximations to the analysis of renewable energy data. In the first application, error estimates gleaned from repeated measurements are incorporated into sparse regression algorithms (LASSO and the Dantzig selector) to minimize the statistical uncertainty of the resulting model. Applied to real biomass data, this approach leads to sparser regression coefficients corresponding to improved accuracy as measured by k-fold cross validation.
error. In the second application, a regression model based on separated representations is fit to reliability
data for cadmium telluride (CdTe) thin-film solar cells. The data is inherently multi-way, and our approach
avoids artificial matricization that would typically be performed for use with standard regression algorithms.
Two distinct modes of degradation, corresponding to short- and long-term decrease in cell efficiency, are
identified. In the third application, some theoretical properties of a popular chemometrics algorithm called
orthogonal projections to latent structures (O-PLS) are derived.
Dedication

To my parents, Madalyn and Bob
Acknowledgements

I would like to thank my adviser Greg Beylkin for his constant availability, curiosity, and insight, all of which have greatly augmented my experience as a graduate student and researcher. Alireza Doostan has also offered a great deal of encouragement and expertise, and I look forward to working with him in the coming year. At NREL, my adviser Peter Graf has been a consistent source of inspiration both within and outside of the research environment. Wes Jones, in addition to always finding a way to keep me fed for the last few years, has always generated a constant stream of healthy and pointed scientific debate that is a benefit to all who engage it.

I’m also grateful for the interaction with a number of other collaborators and colleagues in the last few years including Lucas Monzon, Ryan Elmore, Ross Larsen, Mike Sprague, Christopher Chang, Changwon Suh, Dave Astling, Dave Albin, and Rebekah Graham.

I learned a lot early on as a graduate student, both on and off campus, from Terry Haut and Geoff Sanders. The path was also shared in one way or another with a number of other current and former grad students including Adam Fox, Seb Skardal, Ted Galanthay, Dan Larremore, Dan Kaslovsky, Dane Taylor, Jason Hammond, Brock Mosovsky, Henry Romero, Darren Homringhauser, Kristine Snyder, Jenny Russell, Justin Kuczynski, Erin Byrne and Ryan Wilson.

Craig Faulhuber and Mike Levy have been consistent partners in life outside of research: I hope that tradition continues. My newer friends Natalie Jaime, Mark Jaeger, Jenny Key, Will Hunt and Carrie Molloy have also been good for a frequent laugh and occasional climbing adventure. Sarah Boman sat patiently and supportively through the final pangs of my graduate career while being a graduate student herself, and has always been up for stress-destroying adventures when time permits: thank you!
Going back even further, I should acknowledge Ryan Smith, Maria Belmonte-Rivas and John and Pam Bowlan as being early compatriots in math and physics, patient friends, and general life enthusiasts.

Last but not least, thanks to my family for their constant support and encouragement. My parents Madalyn and Bob have played no small role in my education and work ethic which have been priceless to me. My brothers Mark and Steve, my oldest friends, have also been great companions and role models along the way. I hope I can return the favor in the years to come.
Chapter

1 Numerical construction of Green’s functions in high dimensional elliptic problems with variable coefficients

1.1 Introduction

1.1.1 Curse of dimensionality

1.1.2 Ill-conditioning

1.1.3 Multiple or evolving source terms

1.2 Review of canonical tensor decompositions

1.2.1 Motivation for their use

1.2.2 Basic operations

1.2.3 Ill-posedness (and why it’s acceptable)

1.2.4 Reducing the separation rank

1.3 The discrete problem in separated format

1.3.1 Separability of the constant coefficient Green’s function

1.3.2 Separated format of the differential operator

1.3.3 Symmetric second order discretization

1.4 Construction of the Green’s functions

1.4.1 Schulz iteration for constructing the inverse

1.4.2 Enforcing sparsity

1.4.3 Preconditioning with the constant coefficient Green’s function
A.2 Regression under design uncertainty ........................................... 58
  A.2.1 Model ................................................................. 59
  A.2.2 The challenge of design uncertainty ................................. 60
A.3 Scaling penalizes design uncertainty in the solution path .......... 62
  A.3.1 A prototypical pursuit algorithm: forward stagewise (FS) regression .................. 62
  A.3.2 Main result .................................................................. 63
  A.3.3 Connection to the Dantzig selector ................................ 66
A.4 Application to biomass characterization data ....................... 67
  A.4.1 Description of the data .................................................. 68
  A.4.2 Methods ................................................................. 70
  A.4.3 Results ................................................................. 70
  A.4.4 Discussion .............................................................. 73
A.5 Conclusions ................................................................. 73

B Analysis of Governing Factors for Photovoltaic Loss Mechanism of n-CdS/p-CdTe Heterojunction via Multi-way Data Decomposition 78
  B.1 Introduction ............................................................... 79
  B.2 Experimental ............................................................. 80
    B.2.1 Fabrication and measurement of CdS/CdTe cells .................. 80
    B.2.2 Difficulty in visualizing global trends ............................. 81
  B.3 Quantitative Methodology: N-way Regression ..................... 83
    B.3.1 Pattern analysis ...................................................... 83
    B.3.2 Formulation of N-Way Partial Least Square (PLS) algorithm .... 84
  B.4 Results and Discussion ................................................ 86
    B.4.1 The data in N-way format .......................................... 86
    B.4.2 Data pre-processing ................................................. 86
    B.4.3 Model selection ...................................................... 87
Preface

This thesis consists of two parts, corresponding to two distinct lines of research undertaken by the author over the last several years. Part I (Chapters 1 and 2) is focused on the efficient numerical solution of a class of elliptical partial differential equations in high dimensions, and was done in collaboration with Greg Beylkin and Alireza Doostan at CU Boulder. While we plan to refine and submit Chapters 1 and 2 as independent publications, so far they only appear in this dissertation. Accordingly, we provide motivation and context for this work here in the thesis introduction.

Part II (Appendices A-C) center on the analysis of real-world data generated in renewable energy applications, and was done while working as a research assistant in the Computational Sciences Center at the National Renewable Energy Laboratory (NREL). Collaborators on these projects include Peter Graf, Wes Jones, Ryan Elmore, Dave Astling, Changwon Suh, Dave Albin, and Rebekah Graham. These chapters are added as Appendices as they have already been submitted and are either currently under review (Appendices A-B) or have already been published (Appendix C). As such, the motivation and background for these chapters is well developed and may be found therein.

In Part I of this thesis, we consider solving boundary value problems (BVP’s) of the type,

\[
\begin{aligned}
\mathcal{A}u &= f \text{ in } \mathcal{D}, \\
u &= g \text{ on } \partial \mathcal{D},
\end{aligned}
\]

where \( \mathcal{A} \) is an elliptic operator, \( \mathcal{D} \) is a domain in \( \mathbb{R}^d \), and \( \partial \mathcal{D} \) is its boundary. Of particular interest is the variable coefficient Poisson equation

\[
\begin{aligned}
-\nabla \cdot (a(x) \nabla u(x)) &= f(x), \quad x \in \mathcal{D}, \\
u(x) &= g(x), \quad x \in \partial \mathcal{D},
\end{aligned}
\]  

(1)

where \( \nabla = \sum_{i=1}^{d} (\partial / \partial x_i) e_i \), with \( e_i \) the \( i \)th standard basis vector. This equation appears in a range of applications, including electrostatics, structural mechanics, thermodynamics, quantum mechanics and chromodynamics.

Techniques for solving (1) may be divided into two basic types: iterative and direct. In iterative methods, the BVP is typically represented by an \( N \times N \) linear system corresponding to \( N \) degrees of
freedom in the discretization. The iteration of choice, often conjugate gradient or GMRES [120], is then performed to obtain the solution for a given right hand side. The iteration converges at a rate dependent on the conditioning of the system - known in this setting to be poor - and must be performed as many times as the right hand side changes. Numerous methods tailored specifically to the Poisson and Poisson-Boltzmann equations, utilizing finite differences, finite elements, and finite volumes, have been and continue to be proposed [146, 40, 12, 57, 107, 34].

In the direct approach, discretization occurs within the integral (as opposed to differential) formulation of the BVP. Formally, an approximate Green’s function is constructed as well as an appropriate quadrature rule for application to the right hand side. The integral formulation has the advantage of being well conditioned because it comes down to the application of a bounded operator (as opposed to solving with an ill-conditioned one). So long as the Green’s function is fast to apply, the framework is thus efficient for multiple right hand sides. However, using classical techniques such as Gaussian elimination, constructing the inverse operator requires $O(N^3)$ operations with a nominal cost of $O(N^2)$ to apply. To address this issue, many fast algorithms have been developed for various problems, bringing the cost of the construction to $O(N^a \log^b N)$ operations with $1 \leq a \leq 2$ and $0 \leq b \leq 2$, and cost of application $O(N)$ or $O(N \log N)$ (see, e.g., [62]).

Some of the most successful integral methods center on the compression of low-rank blocks of the discretized system. These methods include fast multipole (FMM) [63, 38], panel clustering [64, 121], $\mathcal{H}$-matrices [65, 67], matrix skeletonization [99, 37], and hierarchically semiseparable (HSS) matrices [33, 32, 149, 98]. In all of these approaches, the inverse operator is mapped to a (typically dense) $N \times N$ matrix. However, blocks of the dense matrix corresponding to “far field” interactions tend to be low-rank, allowing for their compression by, e.g., column skeletonization or the singular value decomposition (SVD). Thus, despite the nominal density of the full matrix, compression of the low-rank blocks leads to a sparse representation that is fast to apply.

A major difference between the approach we adopt and those mentioned above is that, rather than forming a linear system explicitly, we work directly with the tensor equations in a multilinear format. Specifically, we consider problems in which the functions involved are well represented by sums of products of
univariate functions,

\[ f(x_1, x_2, \ldots, x_d) = \sum_{l=1}^{r} \prod_{j=1}^{d} f_j^{(l)}(x_j) + O(\epsilon) \]

(2)

with error \( \epsilon \) and separation rank \( r \). By adhering to the low separation rank assumption, we of course limit the scope of the approach to certain settings. One the other hand, as we describe in Chapter 1, working in a tensor product format allows us to invoke certain powerful analytical results regarding the sparsity of the Green’s functions. Of even greater significance is the fact that, formally, the complexity of the representation scales linearly in the dimension, \( d \). This is particularly useful in the context of solving stochastic ordinary/partial differential equations (SODE’s/SPDE’s): in such problems, the presence of random inputs can lead lead to rapid growth in the formal dimension [150, 80, 46].

When computing in high dimensions, one must address the exponential growth of parameters: the so-called “curse of dimensionality” (first attributed to Bellman in [1]. See also Donoho’s lecture [44] for an interesting point of view on the subject). Simply put, many classical approaches simply cannot be used when \( d \gg 1 \). For instance, consider a spectral method that represents a function of interest in terms of \( M \) orthonormal basis functions,

\[ f(x) = \sum_{m=1}^{M} \left\langle f, \varphi^{(m)} \right\rangle \varphi^{(m)}(x), \]

with \( \langle \cdot, \cdot \rangle \) an appropriately defined inner product. Now suppose that, in dimension \( d \), the multivariate function \( f(x_1, x_2, \ldots, x_d) \) is decomposed via a tensor product of orthonormal bases \( \{ \varphi_j^{(m)} \}_{m=1}^{M} \) of each dimension:

\[ f(x_1, x_2, \ldots, x_d) = \sum_{m_1=1}^{M} \sum_{m_2=1}^{M} \cdots \sum_{m_d=1}^{M} \alpha_{m_1, m_2, \ldots, m_d} \varphi_1^{(m_1)} \varphi_2^{(m_2)} \cdots \varphi_d^{(m_d)}, \]

\[ \alpha_{m_1, m_2, \ldots, m_d} = \left\langle f, \varphi_1^{(m_1)} \varphi_2^{(m_2)} \cdots \varphi_d^{(m_d)} \right\rangle \]

The need to compute all \( M^d \) projection coefficients makes this approach intractable for even mild \( M \) and \( d \).

Starting with the independent work of Smolyak and Babenko in the early 1960’s [10, 125], a number of “sparse grid” approaches have been developed in an effort to address the curse of dimensionality (see, e.g., [26]). The methods employ tensor product grids that utilize fewer than \( M^d \) combinations via various constraints and decay arguments. Nevertheless, the number of free parameters needed is \( O(M(\log M)^{d-1}) \) and thus still exponential in the number of grid points, making such approaches useful only for moderately larger dimensions than allowable using full tensor products.
In contrast, consider the discretization of the separated representation (2) using \( M \) equispaced points in each dimension,

\[
\mathcal{F}_{i_1,i_2,...,i_d} = \sum_{l=1}^{r} \prod_{j=1}^{d} f_{ij}^{(l)} \quad \text{or} \quad \mathcal{F} = \sum_{l=1}^{r} \bigotimes_{j=1}^{d} f_{ij}^{(l)}.
\]

Here the symbol \( \otimes \) signifies the tensor (outer) product of vectors \( f_{ij}^{(l)} \) with elements \( f_{ij}^{(l)} \) where \( i_j = 1, \ldots, M \) for all \( j = 1, \ldots, d \). Evidently, the discretized tensor is parametrized by \( d \cdot r \cdot M \) numbers which, so long as \( r \) is sufficiently small, represents a dramatic reduction compared with tensor and sparse grid techniques.

Of course, this begs the question: “which types of functions admit a low-separation rank representation?” Unfortunately, few theoretical results exist to answer this question in general terms. But even so, for the types of problems we consider here, many of the functions and operators of interest will satisfy the low-rank requirement.

The use of the tensor format (3), referred to in the literature as both a canonical decomposition (CTD or CANDECOMP) and parallel factorization (PARAFAC), has been used for high-dimensional problems of many kinds. These include applications in statistics, psychometrics, chemometrics, quantum chemistry, signal processing, numerical linear algebra, and many others (see, e.g., [11] and references therein for a good overview).

The canonical decomposition is by no means the only one. For example, the Tucker decomposition, also knowns as the high-order SVD (HOSVD) [138, 88, 41], is intended to generalize the product structure of the matrix SVD

\[
A = U \Sigma V^*.
\]

where \( U \in \mathbb{R}^{m \times r} \) and \( V \in \mathbb{R}^{n \times r} \) are unitary, \( \Sigma \in \mathbb{R}^{r \times r} \) is a diagonal matrix with diagonal elements \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0 \), and \( r \) is the matrix rank. In the \( d \)-dimensional case, there are matrices analogous to \( U \) and \( V \) in each dimension and a \( d \)-dimensional “core” tensor analogous to \( \Sigma \). Unfortunately, the number of parameters in the HOSVD is also exponential in \( d \), and so does not appear to be a viable decomposition for general high dimensional problems.
A more recent decomposition is the tensor train (TT) [110] which is intended to scale like the canonical decomposition while avoiding some of its theoretical shortcomings (discussed further in Section 1.2.3). In this format,

$$F_{i_1, i_2, \ldots, i_d} = G^{(1)}_{i_1} G^{(2)}_{i_2} \cdots G^{(d)}_{i_d},$$

where $G^{(k)}_{i_k}$ is an $r_{k-1} \times r_k$ matrix for each of the indices, $i_k = 1, \ldots, M_k$, with $r_0 = r_d = 1$ to ensure that (5) is scalar. If the TT-ranks satisfy $r_k \leq r$ and $M_k \leq M$, then the tensor train format is stored in $O(d \cdot r^2 \cdot M)$ numbers. While this decomposition has shown some promise in numerical applications [111], it is not physically intuitive and is disconnected from a body of analytical results that, when working in the separated representation, may be exploited to great advantage.

The canonical tensor has been used extensively in the context of numerical partial differential equations (e.g., [103, 104, 68, 69, 70, 80, 86, 66] and the references therein). Our contribution in this thesis is two-fold:

1. we address the case of variable coefficients, and
2. we describe a fast, randomized approach to rank reduction that increases the separation rank of the operators that may be considered.

These two contributions constitute the first two chapter of the thesis.
Chapter 1

Numerical construction of Green’s functions in high dimensional elliptic problems with variable coefficients
1.1 Introduction

In this chapter, we describe an efficient algorithm for approximating the Green’s function of the variable coefficient Poisson equation,

\[
\begin{aligned}
-\nabla \cdot (a(x) \nabla u(x)) &= f(x), \quad x \in D \\
u(x) &= g(x), \quad x \in \partial D
\end{aligned}
\tag{1.1}
\]

in arbitrary dimension \(d\). Unless otherwise stated, we assume that the functions \(a_j^{(l)}\) are at least once differentiable so that the quantity \(-\nabla \cdot (a \nabla u)\) may be formally expanded. We further assume that the tensor-valued coefficient is separable with separation rank \(r\),

\[
a(x_1, x_2, \ldots, x_d) = \sum_{l=1}^{r} \sigma_l a_1^{(l)}(x_1)a_2^{(l)}(x_2) \cdots a_d^{(l)}(x_d) + O(\epsilon),
\]

or more compactly,

\[
a(x) = \sum_{l=1}^{r} \sigma_l \prod_{j=1}^{d} a_j^{(l)}(x_j) + O(\epsilon).
\tag{1.2}
\]

Here, the weights \(\sigma_l\) are chosen so that the univariate functions have unit \(L_2(\mathbb{R})\) norm,

\[
||a_j^{(l)}||_2^2 = \int_{X_j} |a_j^{(l)}(x_j)|^2 dx_j = 1,
\]

and \(\epsilon\) is a user-supplied accuracy. In numerical applications, \(r\) typically depends on the desired accuracy.

Remark. The decomposition (1.2) goes by several names, including canonical tensor decomposition (CTD) and parallel factorization (PARAFAC). In this thesis, we will refer to such decompositions as canonical.

In general, we do not require that the univariate functions come from any specific class as they may be computed numerically “on the fly” as described in Section 1.2.4. In order for our approach to be tractable, however, we do require that the functions and operators of interest have small separation rank (we will elaborate on this later). Thus, it may be helpful to think of the variable coefficient as having a short Fourier series or, alternatively, a lengthy Fourier series with only a few non-zero projection coefficients.

Approximating a solution of (1.1) is challenging from several points of view which, by way of introduction, we outline next.
1.1.1 Curse of dimensionality

In order for any numerical method to be feasible in high dimensions, it must be able to confront the so-called “curse of dimensionality.” As an illustration, consider approximation of a function \( f \) in using a spectral basis in dimension \( d \), with \( M \) basis functions in each dimension. Classically, this leads to expressions of the type

\[
f(x_1, x_2, \ldots, x_d) = \sum_{m_1=1}^{M} \sum_{m_2=1}^{M} \cdots \sum_{m_d=1}^{M} \alpha_{m_1, m_2, \ldots, m_d} \phi_1^{(m_1)} \phi_2^{(m_2)} \cdots \phi_d^{(m_d)}
\]

with \( M^d \) linear projection coefficients. Even mild choices of the parameters \( d, M \) leads to an intractable problem (e.g., by some estimates the number \( 10^{30} \) is roughly the number of kilometers in the observable universe). On the other hand, in such cases it may be physically obvious that the number of degrees of freedom is not so great. The task then becomes one of finding the appropriate mathematical description of the functions of interest via another (non-linear) approximation scheme. For us, canonical decompositions as in (1.2) will be essential in overcoming the “curse.”

1.1.2 Ill-conditioning

Consider the discretized system of equations corresponding to (1.1),

\[
Au = f
\]

(1.3)

Such a system will typically be sparse, allowing for the rapid application of \( A \). However, it is also ill-conditioned on account of the unboundedness of the differential operator. In particular, for a uniform discretization of step size \( h \), \( A \) has condition number \( \kappa = O(1/h^2) \). Given that typical iterative methods for solving (1.3) have convergence rates of \( O(\sqrt{\kappa}) \), such ill-conditioning can become the central obstacle to efficiently approximating \( u \).

An alternative formulation, and the one that we adopt, is to cast the problem as an integral equation,

\[
u(x) = \int_{\Omega} G(x, y)f(y)dy,
\]

(1.4)
where the kernel $G(x, y)$ is the Green’s function associated with the differential operator and corresponding boundary conditions. Given the Green’s function and a quadrature rule for evaluating the integral (1.4), the problem takes on the form

$$u = Gf,$$  \hspace{1cm} (1.5)

where evidently $G = A^{-1}$, and $u$ is evaluated by a single matrix-vector multiply.

Given a discretized Green’s function, the act of computing $u$ is very well conditioned: the operator is bounded, and one has only to apply it to the right hand side (as opposed to solving in the differential formulation). In this chapter, we will describe how to:

- compute the Green’s function in a stable, well conditioned way and

- ensure its rapid application.

This last point is important because, in general, the Green’s function will be dense, leading to $O(N^2)$ cost to apply. To address this, we exploit the fact that both the differential and integral operators described above are sparse in a wavelet basis (see, e.g., [17, 15]).

### 1.1.3 Multiple or evolving source terms

In many applications, solutions may be sought corresponding to multiple source terms. Consider, for example, modeling the diffusion of electrons in a semiconductor with an electro-motive force defined by the coupled system:

$$\begin{cases} 
- \nabla \left( \varepsilon(x) \nabla \phi(x, t_k) \right) = \rho(x, t_k) \\
F_i(x_i(t_k)) = -q_i \nabla \phi(x(t_k)) \\
\rho(x, t_k) = \sum_{i=1}^{N} q_i \delta(x - x_i(t_k)) 
\end{cases}$$

Here $\varepsilon$ is the (time-independent) permittivity, $\rho$ is the volume charge distribution, $\phi$ is the electrostatic potential, $F_i$ is the force on charge $i$, $q_i$ is the charge quantity, and $t_k$ are time steps. At each time step, the electrostatic potential determines the force on the holes and electrons while, on the other hand, the location of the charges determines the source term, $\rho(x, t_k)$. In cases like this, the additional cost of approximating the Green’s function “off-line” may be acceptable so long as it can be applied quickly “on-line” when needed.
1.2 Review of canonical tensor decompositions

The history of canonical decompositions is a long one, going back to the early twentieth century [76, 77]. In the interest of space, we refer the reader to an excellent and fairly recent review of the theory and history of the decomposition (1.2) and several others by Kolda and Bader [11]. Here will simply summarize the key concepts and algorithms needed for our approach.

For simplicity, and without loss of generality (w.l.o.g.), let us assume that all of the univariate functions \( a_j^{(l)}(x_j) \) in (1.2) are defined on the interval \( \tilde{D} \in [0, 1] \) so that \( D = \otimes_{j=1}^{d} \tilde{D} \), and that discretization of such functions is performed uniformly with \( M \) grid points with corresponding step size \( h = 1/(M - 1) \). Thus we may think in terms of vectors in each dimension, with \( [a_j^{(l)}]_i = a_j^{(l)}(x_{ij}), \ x_{ij} = ih, \) for \( i = 0, \ldots, M - 1 \), that combine to give a discrete tensor with \( d \) indices,

\[
a_{i_1i_2\ldots i_d} = \sum_{l=1}^{r} \sigma_l \prod_{j=1}^{d} a_j^{(l)}(x_{ij}), \quad \text{or} \quad \mathbf{a} = \sum_{l=1}^{r} \sigma_l \bigotimes_{j=1}^{d} \mathbf{a}_j^{(l)}, \quad (1.6)
\]

where the tensor \( \mathbf{a} \) is distinguished from the 1D vectors \( \mathbf{a}_j^{(l)} \) by context.

1.2.1 Motivation for their use

To understand the effectiveness of using of CTD’s for numerical applications in high dimensions, first let us consider the Singular Value Decomposition (SVD) in dimension \( d = 2 \). Recall that for any \( m \times n \) matrix \( \mathbf{A} \) of full rank \( r = \min(m, n) \), there exist unitary matrices \( \mathbf{U} \in \mathbb{R}^{m \times r} \) and \( \mathbf{V} \in \mathbb{R}^{n \times r} \) and a diagonal matrix \( \mathbf{\Sigma} \in \mathbb{R}^{r \times r} \) such that

\[
\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^*,
\]

where the diagonal elements of \( \mathbf{\Sigma} \) satisfy \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0 \). Alternatively, one can express \( \mathbf{A} \) as a sum of rank-one matrices,

\[
A_{ij} = \sum_{l=1}^{r} \sigma_l u_i^{(l)} v_j^{(l)}, \quad \text{or} \quad \mathbf{A} = \sum_{l=1}^{r} \sigma_l \mathbf{u}^{(l)} \otimes \mathbf{v}^{(l)}. \quad (1.7)
\]
By the Eckart-Young Theorem [47], the optimal rank-k approximation of \( A \) is given by the first \( k \) terms of its SVD,

\[
\arg\min_{B: \text{rank}(B)=k} \|A - B\|_2 = \sum_{l=1}^{k} \sigma_l u^{(l)} \otimes v^{(l)}.
\]

Now suppose that \( A \) is well approximated by a matrix \( B \) of rank \( k \ll \min(m, n) \) (i.e., \( A \) is a low rank matrix). In the SVD format, \( B \) is described by \( k(m + n) \) numbers (the singular values are constrained by orthonormality of the singular vectors), compared with \( m \cdot n \) for \( A \). For \( k \) sufficiently small, the compression of \( A \) via its truncated SVD is thus exponential.

Comparing with (1.6), we see that the SVD is actually a canonical tensor decomposition in dimension \( d = 2 \) (admittedly, with very special properties). The primary difference is that, for arbitrary tensors in dimension \( d > 2 \), there is no equivalent of the Eckart-Young theorem guaranteeing an optimal series of rank-one tensors (see Section 1.2.3). On the other hand, if a \( d \)-dimensional tensor is well-represented by a CTD of low separation rank, the compression is exponential. To see this, assume that each dimension of a tensor is discretized at \( M \) points. Then full tensor contains \( M^d \) distinct numbers (combinations of indices) while the CTD is determined by only \( r \cdot d \cdot M \).

To our knowledge, there is no complete theory concerning the separation rank of arbitrary functions in high dimensions. However, for many high dimensional operators and functions of interest, one can indeed find a low-rank separated representation for a given accuracy (cf. Section 1.3). In such cases, the usefulness of the separated formulation will be manifest.

### 1.2.2 Basic operations

Here we briefly summarize the basic CTD operations needed for our method.

#### 1.2.2.1 Inner product

The discrete inner product between two arbitrary tensors \( A \) and \( B \) is defined straightforwardly as

\[
\langle A, B \rangle = \sum_{i_1=1}^{M} \cdots \sum_{i_d=1}^{M} A_{i_1, \ldots, i_d} B_{i_1, \ldots, i_d}.
\]
If \( A \) and \( B \) admit CTD’s of rank \( r_A \) and \( r_B \), respectively, then

\[
\langle A, B \rangle = \sum_{l=1}^{r_A} \sigma^A_l \sum_{m=1}^{r_B} \sigma^B_m \prod_{j=1}^d \langle a^{(l)}_j, b^{(m)}_j \rangle,
\]

with the weights distinguished by a superscript.

### 1.2.2.2 Frobenius norm

The most commonly used norm for CTD applications is the Frobenius norm. It may be defined simply in terms of the inner product,

\[
||A||^2_F = \sum_{i_1, \ldots, i_d} A_{i_1, \ldots, i_d}^2 = \langle A, A \rangle.
\]

### 1.2.2.3 Operator multiplication and adjoint

The above operations are defined for arbitrary CTD’s. However, in our application we need to distinguish between functions and operators (in the discretized case, “vectors” and “matrices”). Notationally, we will use capital “blackboard” letters to signify an operator (except for \( \mathbb{R} \), which will always denote the set of real numbers) and lowercase bold letters for simple pointwise functions, distinguished from simple vectors by context. Upper-case, calligraphic fonts such as \( \mathcal{A} \) will be used for generic tensors.

With these conventions, the Poisson equation (1.1) in CTD format is expressed as

\[
\mathcal{A} u = f,
\]

with

\[
\mathcal{A} = \sum_{l=1}^{r_A} \sigma^A_l \bigotimes_{j=1}^d A_{j}^{(l)},
\]

\[
u = \sum_{l=1}^{r_u} \sigma^u_l \bigotimes_{j=1}^d u^{(mi)}_j,
\]

and

\[
f = \sum_{l=1}^{r_f} \sigma^f_l \bigotimes_{j=1}^d f_j^{(l)}.
\]
Table 1.1: Cost of elementary CTD operations.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Definition in CTD format</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inner product</td>
<td>$\langle U, V \rangle = \sum_{l=1}^{r_U} \sum_{m=1}^{r_V} \sigma_l^U \sigma_m^V \prod_{j=1}^{d} \langle u_l^{(j)}, v_m^{(j)} \rangle$</td>
<td>$O(d \cdot r_U \cdot r_V \cdot M)$</td>
</tr>
<tr>
<td>Frobenius norm</td>
<td>$</td>
<td></td>
</tr>
<tr>
<td>Matrix-matrix multiply</td>
<td>$AB = \sum_{l=1}^{r_A} \sum_{m=1}^{r_B} \sigma_l^A \sigma_m^B \prod_{j=1}^{d} A_l^{(j)} B_m^{(j)}$</td>
<td>$O(d \cdot r_A \cdot r_B \cdot M^3)$</td>
</tr>
<tr>
<td>Matrix-vector multiply</td>
<td>$Au = \sum_{l=1}^{r_A} \sum_{m=1}^{r_u} \sigma_l^A \sigma_m^u \prod_{j=1}^{d} A_l^{(j)} u_m^{(j)}$</td>
<td>$O(d \cdot r_A \cdot r_u \cdot M^2)$</td>
</tr>
</tbody>
</table>

Here, each matrix $A_j^{(i)}$ is an $M \times M$ discretized operator in dimension $j$ (we elaborate in Section 1.3.3) and $u_j^{(i)}, f_j^{(i)}$ are $M$-vectors. The operator $A : (\otimes_{j=1}^{d} \mathbb{R}^M) \rightarrow (\otimes_{j=1}^{d} \mathbb{R}^M)$ is described by $2d$ indices,

$$A_{i_1' i_2' ... i_d'} = \sum_{l=1}^{r_A} \sigma_l^A \prod_{j=1}^{d} [A_l^{(j)}]_{i_j i_j'}.$$

Similarly, the product of two operators is given by

$$A B = \sum_{l=1}^{r_A} \sum_{m=1}^{r_B} \sigma_l^A \sigma_m^B \prod_{j=1}^{d} A_l^{(j)} B_m^{(j)},$$

and the adjoint by term-wise conjugation,

$$A^* = \sum_{l=1}^{r_A} \sigma_l^A \prod_{j=1}^{d} (A_l^{(j)})^*.$$

Matrix-vector multiplication defined analogously, for example,

$$f = \sum_{l=1}^{r_A} \sum_{m=1}^{r_u} \sigma_l^A \sigma_m^u \prod_{j=1}^{d} A_l^{(j)} u_m^{(j)}.$$

The nominal cost of the elementary CTD operations are given in Table 1.1. All are linear in $d$.

### 1.2.3 Ill-posedness (and why it’s acceptable)

In two dimensions ($d = 2$), the canonical decomposition has the same form as the SVD. However, the optimality of the SVD guaranteed by the Eckart-Young Theorem [47] is no longer assured for $d > 2$. As an example, widely cited in the literature [42, 90, 87, 112], consider the tensor of nominal separation rank $r = 3$,

$$A = a^{(1)} \otimes b^{(1)} \otimes c^{(2)} + a^{(1)} \otimes b^{(2)} \otimes c^{(1)} + a^{(2)} \otimes b^{(1)} \otimes c^{(1)},$$

where the six vectors satisfy:

$$\langle a^{(1)}, a^{(2)} \rangle, \langle b^{(1)}, b^{(2)} \rangle, \langle c^{(1)}, c^{(2)} \rangle \neq 0.$$
Now consider a rank-two approximation of $\mathcal{A}$,

$$\tilde{\mathcal{A}} = \alpha (a^{(1)} + \frac{1}{\alpha} a^{(2)}) \otimes (b^{(1)} + \frac{1}{\alpha} b^{(2)}) \otimes (c^{(1)} + \frac{1}{\alpha} c^{(2)}) - \alpha a^{(1)} \otimes b^{(1)} \otimes c^{(1)},$$

where $\alpha > 0$. It is straightforward to show that $\tilde{\mathcal{A}} \to \mathcal{A}$ as $\alpha \to \infty$, implying that $\mathcal{A}$ does not have an optimal rank-2 approximation (this phenomenon is a general one in dimension $d > 2$; we refer the reader to [42] for a detailed discussion of this issue).

Despite the ill-posedness of finding rank-optimal approximations, there are several reasons to persist in using the CTD formulation (cf. [102]). The most important for us is that we simply do not need the optimal approximation: we are satisfied so long as the separation rank is reasonably small for a given accuracy. Given that the separated representation leads to $d \cdot r \cdot M$ parameters versus $M^d$ for the full tensor, any moderate separation rank leads to tractability where it might otherwise not exist.

1.2.4 Reducing the separation rank

An important operation needed for working in the separated format is the reduction of the separation rank. The question: “can the tensor $\mathcal{A}$ be approximated by a lower-rank tensor $\mathcal{B}$ to within accuracy $\epsilon$?” is often not answerable analytically. In practice, the answer comes from applying a rank reduction algorithm.

1.2.4.1 Alternating Least Squares (ALS)

The workhorse algorithm for rank reduction is Alternating Least Squares (ALS) (see, e.g., [104], Section 3.1 and [11], Section 3.4). Briefly, ALS sweeps through the $d$ dimensions holding all but one fixed, optimizes it in a least squares sense, and then repeats until the iteration either converges or becomes stuck. If no a priori estimate of the separation rank is available, typically an initial guess with rank one is used. If the ALS iteration becomes stuck, the rank is increased by one and the procedure is repeated.

Let us briefly estimate the cost of performing ALS. Denote by $\mathcal{A} = \sum_{i=1}^{r_A} \sigma_i^A \otimes_{j=1}^d a^{(l)}_j$ the tensor for which a low rank approximation, $\mathcal{B} = \sum_{i=1}^{r_B} \sigma_i^B \otimes_{j=1}^d b^{(l)}_j$, is sought. Consider fixing all but dimension $k$ and optimizing over the $r_B \cdot M$ parameters corresponding to that dimension. For a fixed rank $r_B$, one solves the normal equations,

$$G = HU^{(k)},$$

(1.8)
where

\[ G_{ij} = \prod_{j \neq k} \langle b_j^{(l)}, b_j^{(l')} \rangle, \]

\[ H_{lk} = \sum_{l'=1}^{r_B} \sigma_{lk} a_{lk}^{(l)} \prod_{j \neq k} \langle a_j^{(l')}, b_j \rangle. \]

Solving (1.8) for \( U^{(k)} \) and normalizing \( \sigma_{l}^{B} = \sum_{i=1}^{M} U_{il}^{(k)} \), \( U_{il}^{(k)} \leftarrow U_{il}^{(k)} / \sigma_{l}^{B} \) gives the desired result for dimension \( k \), and the process is repeated for all \( d \) dimensions. Construction of the matrix \( G \) costs \( O(d \cdot r_B^2 \cdot M) \), \( H \) costs \( O(d \cdot r_A \cdot r_B \cdot M) \), and solving the \( r_B \times M \) least squares system is \( O(r_B^2) \). Recalling that \( r_B < r_A \) and given that this must be done \( d \) times, naively we have a total cost of \( O(d \cdot r_B^3 + d^2 \cdot r_A \cdot r_B \cdot M) \). However, the matrix \( G \) does not have to be computed \( d \) times because each set of inner products \( \langle b_i^{(l)}, b_i^{(l')} \rangle \) changes at most once, i.e., after that dimension has been updated. Hence, the total cost of an ALS iteration is rather \( O(d \cdot r_B(r_B^2 + r_A \cdot M)) \) (importantly, still linear in \( d \)).

A well-known pitfall in using ALS is that the representation may develop degeneracies during the course of the iteration. In other words, ill-conditioning may arise in the form of cancellation of near-collinear terms, leading to high condition number in the sense of

\[ \kappa(B) = \frac{\left(\sum_{l=1}^{r} \sigma_{l}^2\right)^{1/2}}{||B||_F}. \]

This number will be large when terms with large weights cancel one another, leading to small norm compared to the sum of squares of the weights themselves. A simple way to avoid ill-conditioning of the low rank approximation is to regularize the the normal equations (1.8). In particular, one may perturb the system by an amount \( \lambda I \), where \( \lambda \) is just slightly larger than machine precision and \( I \) is the \( r_B \times r_B \) identity matrix [104].

### 1.2.4.2 Successive rank-one approximation with relaxation

An alternative to standard ALS, which may be much faster for large problems, is to use an algorithm based on successive rank-one updates. To the best of our knowledge, the rank-one update problem and an algorithm to solve it was originally proposed in [152].
The problem is to find a rank-one tensor, say $X_{ijk} = \lambda x_i y_j z_k$, to minimize $||A - X||_F^2$. Using a Lagrange multiplier formulation with the constraints $||x|| = ||y|| = ||z|| = 1$, one can show that a stationary point exists when

$$
\begin{align*}
\sum_{jk} A_{ijk} y_j z_k &= \lambda x_i, \\
\sum_{ik} A_{ijk} x_i z_k &= \lambda y_j, \\
\sum_{ij} A_{ijk} x_i y_j &= \lambda z_k, \\
\sum_{ijk} A_{ijk} x_i y_j z_k &= \lambda
\end{align*}
$$

(1.9)

for some scalar $\lambda$. One way to find $X_{ijk}$ is simple fixed point iteration similar in flavor to Gauss-Seidel: initialize $X$ randomly subject to the norm constraints, and update one dimension at a time according to the equations (1.9). Note that if a total of $L$ iterations are required for convergence, the total cost for finding a single rank-one update is $O(d \cdot r_A \cdot M \cdot L)$.

A simple algorithm for approximating $A$ using this fixed point iteration is the following:

**Successive rank-one approximation:**

1. Set $A_1 = A$.

2. Do until convergence:
   
   (a) Find a rank-one approximation $X_k$ to $A_k$ via the equations (1.9)

   (b) Update the residual $A_{k+1} \leftarrow A_k - \lambda_k X_k$

In practice, we have observed that the above algorithm may converge quickly for the first few updates and then quite slowly thereafter. Of even greater concern (and perhaps the cause of slow convergence), it is possible for the iteration to lead to highly ill-conditioned representations. To see this, first note that the optimal fixed point of (1.9), call it $X_1$ with eigenvalue $\lambda_1$, satisfies

$$X_1 = \arg \max_{X^*} |\langle A_1, X^* \rangle| = \arg \max_{X^*} \lambda_1.$$
Suppose the pair \( \{X_1, \lambda_1\} \) has been computed, and perform the rank-one update, \( A_2 = A_1 - \lambda_1 X_1 \). For the next iteration, we seek \( \{X_2, \lambda_2\} \) such that

\[
X_2 = \arg \max_{X^*} |\langle A_2, X^* \rangle| = \arg \max_{X^*} \lambda_2.
\]

Notice that

\[
\langle A_2, X_2 \rangle = \langle A_1 - \lambda_1 X_1, X_2 \rangle = \langle A_1, X_2 \rangle - \lambda_1 c_{12},
\]

where \( c_{12} = \langle X_1, X_2 \rangle \). Now suppose

\[
\langle A_1, X_2 \rangle > \lambda_1 (1 + c_{12}),
\]

which is completely possible since there are no orthogonality constraints on \( X_1 \) and \( X_2 \) (i.e., \( c_{12} \neq 0 \) in general). Then

\[
\lambda_2 = \langle A_2, X_2 \rangle > \lambda_1 (1 + c_{12}) - \lambda_1 c_{12} = \lambda_1.
\]

Thus, even though each rank-one update decreases the error in approximating the current iterate, it may do so at the cost of the conditioning of the representation (the weights may continue to grow as the error decreases).

We have also noticed that these issues may be avoided if we periodically “relax” the solution via ALS [16]. In other words, without changing the rank of the current iterate \( A_n \), we use it as an initial guess for \( A \) in the standard ALS iteration and let it sweep the dimensions some fixed number of times. The weights are re-distributed in the process, helping to avoid possible ill-conditioning and further decreasing the error.

Hence, the algorithm we use in practice is:

**Successive rank-one approximation with relaxation:**

1. Set \( A_1 = A \).

2. Do until convergence:

   (a) Find a rank-one approximation \( X_k \) to \( A_k \) via the equations (1.9).

   (b) Update the residual \( A_{k+1} \leftarrow A_k - \lambda_k X_k \).
(c) If \( \text{mod}(k, m) = 0 \) for integer \( m \), then relax via \( n \) ALS iterations.

By tuning the parameters \( m \) and \( n \), we can trade number of terms for accuracy at each step, thus providing a flexible algorithm. This is particularly useful when the tensor being considered is large and the cost of performing ALS for every separation rank is prohibitive.

### 1.2.4.3 Randomized rank reduction

In Chapter 2, we describe a novel approach for rank reduction using a randomized algorithm which is particularly useful when the separation rank is large and the representation is redundant (i.e., many of the terms are linearly dependent). The need for such an algorithm will become apparent in Section 1.4.1 when we describe the iterative method used to approximate the Green’s function. For now, suppose that we wish to form the self-adjoint operator \( \hat{A}^* \hat{A} \), nominally of rank \( r_A^2 \). The quadratic dependence on the original separation rank is, on the one hand, somewhat artificial: we observe that in many cases the rank can be reduced with sufficient accuracy to \( O(r_A) \) for a relatively mild constant. On the other hand, attempting to reduce the rank by ALS or successive rank-one updating will be prohibitively slow for even modest values of \( r_A \) due to the cost of computing norms and forming the normal equations (cf., Sections 1.2.2 and 1.2.4.1). Thus, as needed, we may first perform a randomized reduction step on the target tensor before using either of those methods.

### 1.3 The discrete problem in separated format

As our method depends on a low-separation rank assumption, we begin by stating some observations and results concerning the constant coefficient Green’s function in separated format. Applying similar arguments to the variable coefficient case, we provide justification for why we expect the Green’s function in this setting to be of manageable separation rank so long as the coefficient is such. We then describe the construction of the variable coefficient differential operator in this format, leading to separation rank at most \( r_a \cdot d \).
1.3.1 Separability of the constant coefficient Green’s function

We begin by stating a key theoretical result developed in [19, 20]. It essentially says that potential-like functions $\rho^{-\alpha}$, where $\alpha > 0$, are well represented by sums of exponentials with near-optimal (minimal) numbers of parameters.

Theorem 1. (Beylkin and Monzon) [20]. For any $\alpha > 0$, $0 < \gamma \leq 1$ and $0 < \epsilon \leq \min\{\frac{1}{2}, \frac{\alpha}{8}\}$, there exist positive numbers $p_l$ and $w_l$ such that

$$\left| \rho^{-\alpha} - \sum_{l=1}^{r} w_l e^{-p_l \rho^2} \right| \leq \rho^{-\alpha} \epsilon, \quad \forall \gamma \leq \rho \leq 1$$

with

$$r = \log \epsilon^{-1} \left[ c_0 + c_1 \log \epsilon^{-1} + c_2 \log \gamma^{-1} \right],$$

where $c_k$ are constants that depend only on $\alpha$. For fixed power $\alpha$ and accuracy $\epsilon$, we have $r = O(\log \gamma^{-1})$.

To see how this relates to our problem, let us consider the following concrete example in two dimensions:

$$\begin{cases} 
-\Delta u(x) = f(x), & x \in [0, 1]^2 \\
 u(0, y) = u(1, y) = 0 \\
 u(x, 0) = u(x, 1) = 0 
\end{cases}$$

Using a standard 3-point stencil (second order finite differences), the Fourier transform of the discretized Green’s may be derived analytically to be

$$\hat{G}^{(\text{Dirichlet})}_{k_1, k_2} = \frac{-1/4}{\sin^2 \left( \frac{k_1 \pi}{2(M+1)} \right) + \sin^2 \left( \frac{k_2 \pi}{2(M+1)} \right)}, \quad k_1, k_2 = 1, \ldots, M. \quad (1.10)$$

(See, e.g., [82], Chapter 7. The expression generalizes straightforwardly to $d > 2$).

Setting $\xi_{k_1}^2 = \sin^2(k_1 \pi/2(M + 1))$ and $\xi_{k_2}^2 = \sin^2(k_2 \pi/2(M + 1))$, and $\rho_{k_1, k_2}^2 = \xi_{k_1}^2 + \xi_{k_2}^2$, the Green’s function is seen to be radial in the new coordinates,

$$\hat{G}^{(\text{Dir})}_{k_1, k_2} = -\frac{1}{4\rho_{k_1, k_2}^2}.$$
As this is exactly the type of kernel described by Theorem 1, there exists a low-separation rank representation for the Green’s function (1.10). Upon substituting and transforming to physical space, we have

\[
G^{(\text{Dirichlet})}_{n_1, n_2} = \sum_{l=1}^{r} w_l u^{(l)}_{n_1} u^{(l)}_{n_2},
\]

\[
u^{(l)}_{n_j} = \frac{2}{M+1} \sum_{k_j=1}^{M} e^{-\eta^2 k_j^2} \sin \left( \frac{\pi n_j k_j}{M+1} \right), \quad j = 1, 2,
\]

where \(\{w_l, \eta_l\}_{l=1}^{r}\) are the weights and nodes of the exponential representation.

A similar analysis holds for periodic boundary conditions. In this case, one can show that

\[
G^{(\text{Periodic})}_{k_1, k_2} = \frac{-1/4}{\sin^2 \left( \frac{k_1 \pi}{M} \right) + \sin^2 \left( \frac{k_2 \pi}{M} \right)}, \quad k_j = 0, \ldots, M - 1
\]

\[
G^{(\text{Periodic})}_{n_1, n_2} = \sum_{l=1}^{s} t_l v^{(l)}_{n_1} - \sum_{l=1}^{s} t_l, \quad \text{and}
\]

\[
v^{(l)}_{n_j} = \frac{1}{M} \sum_{k=0}^{M-1} e^{-\chi_l \sin^2 (k_j \pi/M)} e^{2\pi i k j_{n_j}/M},
\]

meaning that the periodic kernel is also nicely separable (here the weight-node pairs are denoted by \(\{t_l, \chi_l\}_{l=1}^{s}\)).

Subtraction by the sum \(\sum_{l=1}^{s} t_l\) removes the singularity associated with \(k_1 = k_2 = 0\), in which case

\[
e^{-\chi_l \sin^2 (k_1 \pi/M) + \sin^2 (k_2 \pi/M)} = e^0 = 1.
\]

Note that the boundary conditions may be mixed in a straightforward fashion by simply combining the one-dimensional Green’s functions in Fourier space and applying the appropriate one-dimensional transforms. For example, the Fourier Green’s function for zero boundary conditions in the \(x-\) and \(y\)-directions and periodic in the \(z\)-direction has the form

\[
G^{(\text{Mixed})}_{k_1, k_2, k_3} = \frac{-1/4}{\sin^2 \left( \frac{k_1 \pi}{2(M+1)} \right) + \sin^2 \left( \frac{k_2 \pi}{2(M+1)} \right) + \sin^2 \left( \frac{k_3 \pi}{M} \right)},
\]

which is transformed analogously with the previous two examples.

### 1.3.2 Separated format of the differential operator

Next we describe separation of the differential operator. The Laplacian operator is manifestly separable with separation rank \(r = d\),

\[
\Delta = \Delta_1 \otimes I \otimes \cdots \otimes I + I \otimes \Delta_2 \otimes \cdots \otimes I + \cdots + I \otimes I \otimes \cdots \otimes \Delta_d,
\]
where \( \Delta_j = \partial^2 / \partial x_j^2 \) for \( j = 1, \ldots, d \) and \( I \) is the identity operator. Less obvious is the fact that the separation rank is further reducible to order \( \mathcal{O}(\log(d||\Delta||\epsilon^{-1})) \) within accuracy \( \epsilon \) [104]. Thus, for a fixed accuracy, the growth of the separation rank is logarithmic in the number of dimensions.

For the variable coefficient problem, we have

\[
\mathcal{A} = -\nabla \cdot (a(x)\nabla)
\]

\[
= -\nabla \cdot \left( \sum_{l=1}^{r_a} \sigma_l^2 \prod_{j=1}^d a_j^{(l)}(x_j) \right) \nabla
\]

\[
= -\sum_{l=1}^{r_a} \sum_{j=1}^d \sigma_l^2 \left( \frac{\partial}{\partial x_j} a_j^{(l)}(x_j) \frac{\partial}{\partial x_j} \right) \prod_{k \neq j} a_k^{(l)}(x_k)
\]

\[
= -\sum_{l=1}^{r_a} \sum_{j=1}^d \sigma_l^2 L_j^{(l)} \prod_{k \neq j} a_k^{(l)}(x_k),
\tag{1.11}
\]

where \( L_j^{(l)} \equiv \frac{\partial}{\partial x_j} a_j^{(l)}(x_j) \frac{\partial}{\partial x_j} \). Notice that this series has nominal separation rank \( r_a \cdot d \). While the expression (1.11) looks somewhat complicated, it is simply the sum over all terms of operators of the form \( L_j^{(l)} \) in one dimension and pointwise multiplication by \( a_k^{(l)} \) in the others. In two dimensions, for example, it is simply

\[
\mathcal{A} = \sum_{l=1}^{r_a} \sigma_l^2 \left( L_1^{(l)} a_2^{(l)}(x_2) + a_1^{(l)}(x_1)L_2^{(l)} \right).
\]

### 1.3.3 Symmetric second order discretization

There are many possibilities for discretizing the operator \( \mathcal{L} \). For concreteness and ease of discussion, we will consider a symmetric, second order discretization on a uniform, staggered grid. First, we describe the construction of the one dimensional operators. We then proceed to construct a \( d \)-dimensional operator in separated format from its one dimensional parts.

Consider the 1D problem on \( x \in [0, 1] \) with zero boundary conditions. Let \( x_i = ih \), with \( i = 0, \ldots, M + 1 \) and \( h = 1/(M + 1) \), and denote the values of \( a \) and \( u \) at these points by \( a_i \) and \( u_i \). We first discretize the quantity \( a(x)u'(x) \) using a second-order finite difference stencil for the first derivative, on the half grid \( x_{i-1/2} = x_i - h/2 \). This gives

\[
a_i u_i' \approx \frac{1}{h} a_i (u_{i+1/2} - u_{i-1/2}).
\]

Applying the same scheme for \( (a_i u_i')' \) yields
\[
(a_i u_i')' \approx \frac{1}{h^2} \left( a_{i+1/2} (u_{i+1} - u_i) - a_{i-1/2} (u_i - u_{i-1}) \right)
= \frac{1}{h^2} \left( a_{i-1/2} u_{i-1} - (a_{i-1/2} + a_{i+1/2}) u_i + a_{i+1/2} u_{i+1} \right).
\]

In matrix format for 1D, recalling that \( u_0 = u_{M+1} = 0 \), we have

\[
Lu = -h^2 f,
\]

\[
\begin{bmatrix}
-(a_1 + a_2) & a_2 & 0 & \cdots & 0 & 0 & 0 \\
a_2 & -(a_2 + a_3) & a_3 & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & a_{M-\frac{1}{2}} & -(a_{M-\frac{1}{2}} + a_{M-\frac{1}{2}}) & a_{M-\frac{1}{2}} \\
0 & 0 & 0 & \cdots & 0 & a_{M-\frac{1}{2}} & -(a_{M-\frac{1}{2}} + a_{M+\frac{1}{2}}) \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
u_3 \\
\vdots \\
u_{M-1} \\
u_M 
\end{bmatrix}
= -h^2
\begin{bmatrix}
f_1 \\
f_2 \\
\vdots \\
f_M 
\end{bmatrix}.
\]

As with other standard finite difference schemes, periodic boundaries may be treated with just a slight modification of \( L \).

The construction of the multi-dimensional operator is now straightforward. For each dimension \( j = 1, \ldots, d \) and term \( l = 1, \ldots, r_a \), define \( L_j^{(l)} \) as above (that is, replace \( a_i \) by \( [a_{j}^{(l)}]_i \)). In addition, let \( A_j^{(l)} = \text{diag}(a_j^{(l)}) \). Then simply replace \( L_j^{(l)} \) with \( L_j^{(l)} \) and \( a_k^{(l)} \) with \( A_k^{(l)} \) in equation (1.11). In two dimensions, for example, we have

\[
A = \sum_{l=1}^{r_a} \sigma_i a_i \left( L_1^{(l)} \otimes A_2^{(l)} + A_1^{(l)} \otimes L_2^{(l)} \right).
\]

### 1.4 Construction of the Green’s functions

#### 1.4.1 Schulz iteration for constructing the inverse

Our goal is to approximate a Green’s function of the elliptic equation,

\[
-\nabla \cdot (a(x) \nabla u(x)) = f(x),
\]
or in CTD format,

\[ Au = f, \]

with appropriately specified boundary conditions. That is, we seek a function \( G(x, y) \) such that

\[ u(x) = \int_{\Omega} G(x, y)f(y)dy, \]

or, in the discrete version, a CTD operator \( G \) such that

\[ u = Gf, \]

where \( ||G - A^{-1}||/||A^{-1}|| \leq \epsilon. \)

The numerical method we use to approximate \( G \) is the Schulz (or Newton-Schulz) iteration \([122, 13, 14, 115]\),

\[ G_{n+1} = 2G_n - G_nAG_n, \]

\[ G_0 = \alpha A^*, \quad (1.13) \]

with \( \alpha \) chosen so that \( ||E_0|| = ||I - G_0A|| < 1. \) The iteration is essentially the tensor version of Newton’s method for the inverse. Recall for scalar functions that Newton’s iteration is given by

\[ x_{n+1} = x_n - h(x_n)/h'(x_n). \]

Defining \( h(x) = \frac{1}{x} - a \) with root \( x = a^{-1} \), Newton’s iteration is then

\[ x_{n+1} = x_n - \frac{1/x_n - a}{-1/x_n^2} = 2x_n - ax_n^2, \]

the scalar analogy of (1.13).

Advantages of using Schulz iteration are as follows:

(1) *Locally quadratic convergence.* To see that the iteration is quadratically convergent, one does not need to invoke Newton’s method at all. Rather, define the residual at iteration \( n \) by,

\[ E_n = I - G_nA, \]
where $I$ is the identity operator, and substitute

\[
E_{n+1} = I - (2G_n - G_n A G_n)A
= I - 2G_n A - (G_n A)^2
= E_n^2.
\]

(1.14)

Having selected $\alpha$ in (1.13) such that $||E_0|| < 1$, we have by induction that $||E_{n+1}|| \leq ||E_0||^{2^n}$.

(2) **Self-correcting.** As in the scalar Newton’s method, small errors due to round-off, truncation of small terms, or other perturbations will not cause the method to diverge. (See, e.g., [71], in which the authors examine the effect of truncation).

(3) **Number of iterations required is $O(\log \kappa(A))$.** Compared with many standard iterations used to solve elliptic problems are $O(\sqrt{\kappa(A)})$ (e.g., conjugate gradient), this is a significant improvement. A more detailed discussion of convergence properties of the iteration may be found in [13, 14, 114, 115]. For the interested reader, we briefly sketch the proof here.

**Proof sketch for convergence in $O(\log(\kappa(A)))$ number of iterations.** Let the SVD of an $M \times M$ matrix $A$ of rank $r$ be given by $U \Sigma V^*$. Set $G_0 = \alpha A^*$, with $\alpha = 1/\sigma_1^2$ where $\sigma_1$ is the largest singular value of $A$. It can be shown that any iterate $G_n$ has an SVD of the form $V \Xi_n V^*$. Thus, $G_n A = VR_n V^*$, where $R_n = \Xi_n \Sigma = \text{diag}\{\rho_1^{(n)}, \rho_2^{(n)}, \ldots, \rho_M^{(n)}\}$.

Now consider the singular values of the error matrix, $E_n = I - G_n A = V(I - R_n)V^*$ (recall that $V$ is unitary, so $V^*V = VV^* = I$), which are given by $1 - \rho_j^{(n)}$, for all $j = 1, \ldots, M$ and iterations $n$. Given the quadratic convergence (1.14), we have

\[
1 - \rho_j^{(n)} = (1 - \rho_j^{(n-1)})^2 = (1 - \rho_j^{(0)})^{2^n} = \left(1 - \frac{\sigma_j^2}{\sigma_1^2}\right)^{2^n}.
\]
Notice that the convergence will be slowest for the smallest eigenvalue, which we denote \( \sigma_r \). Expanding the the binomial expression on the right hand side to leading order in \( \sigma_r^2/\sigma_1^2 = \kappa^{-2} \), we have

\[
(1 - \kappa^{-2})^{2n} = \sum_{j=0}^{2n} \binom{2n}{j} (-\kappa^{-2})^j
= 1 - 2^n \kappa^{-2} + O(\kappa^{-4}).
\]

Notice that to achieve \( 1 - \rho_j^{(n)} \leq 1/2 \) requires about \( n = 2 \log \kappa \) steps. Now let \( m \) be the smallest integer such that \( \epsilon \leq (1/2)^{2m} \). Then another \( \log_2 \log_2 \epsilon^{-1} \) steps are required to push the error from \( 1/2 \) down to at most \( \epsilon \), for a total of approximately \( 2 \log_2 \kappa + \log_2 \log_2 (1/\epsilon) \).

The major disadvantage of using Schulz with a sparse input is that the intermediate and final iterates may be dense. We address this issue in the next section.

### 1.4.2 Enforcing sparsity

The differential operator \( A \) is sparse in a standard basis. Its approximate inverse, \( G \), on the other hand, is dense, making the Schulz (or any other) iterative method prohibitively expensive for large problems.

However, for the type of elliptic operators we consider, the inverse is known to be sparse when represented in a wavelet basis (see, e.g., [17, 15]). In addition, fast algorithms are readily available to compute both the standard and non-standard (NS) forms. We refer the reader to classic literature on wavelets and their applications for a full discussion of these decompositions [96, 39, 101]. For reference, we summarize key operations involving wavelets and wavelet-transformed operators in Table 1.2.

The fact that both \( A \) and \( G \) are sparse in a wavelet basis suggests that we perform the Newton-Schulz iteration in such a space. While the number of non-zeros may try to grow as the iteration proceeds, the self-correcting nature Newton-Schulz allows us to truncate small entries and small terms without significantly affecting convergence (c.f., [71]). Thus, at the cost of a small and controllable amount of accuracy, sparsity can be maintained throughout the inversion procedure.
### Standard form

<table>
<thead>
<tr>
<th>Operation</th>
<th>Order of complexity</th>
<th>Order of non-zeros</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute ( A_w )</td>
<td>( M \log M )</td>
<td>( M \log M )</td>
</tr>
<tr>
<td>Apply ( A_w ) to vector</td>
<td>( M \log M )</td>
<td>( M \log M )</td>
</tr>
<tr>
<td>Apply ( A_w ) to another matrix in standard form</td>
<td>( M \log^2 M )</td>
<td>( M \log M )</td>
</tr>
</tbody>
</table>

### Non-standard (NS) form

<table>
<thead>
<tr>
<th>Operation</th>
<th>Order of complexity</th>
<th>Order of non-zeros</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute ( A_w )</td>
<td>( M \log M )</td>
<td>( M )</td>
</tr>
<tr>
<td>Apply ( A_w ) to vector</td>
<td>( M )</td>
<td>( M )</td>
</tr>
<tr>
<td>Apply ( A_w ) to another matrix in NS-form</td>
<td>( M )</td>
<td>( M )</td>
</tr>
</tbody>
</table>

Table 1.2: Cost of various operations in standard and non-standard (NS) form, including the order of resulting non-zeros. Here \( A \) is understood to be an \( M \times M \) discretized Laplacian and \( A_w \) is its representation in a wavelet basis. The asymptotic constant depends on level of discretization and choice of basis function.

Computing the standard form is completely straightforward using standard, 1D wavelet transforms. As such, our prototype software uses this representation. However, with a little additional work, the code can be modified to use the NS form. The advantage is a factor of \( \log M \) in storage and a factor of \( \log^2 M \) in speed of matrix-matrix multiplication.

#### 1.4.3 Preconditioning with the constant coefficient Green’s function

The convergence rate and accuracy of the Schulz iteration are intrinsically connected to the condition number \( \kappa(A) \), which is known in this setting to be \( O(1/h^2) \). To address this, we propose preconditioning by the Green’s function of the constant coefficient problem. The preconditioner can either be constructed numerically using Schulz iteration or, in some cases, constructed analytically (see Section 1.3.1).
Figure 1.1: Differential and inverse operators in standard and wavelet bases. Upper left: Second order finite difference matrix in standard basis. Upper right: Color-coded values of the inverse operator on log\(_{10}\) scale in the standard basis. The vast majority of the values are within a factor of 10\(^{-6}\) of each other in magnitude. Lower left: Sparsity structure of the differential operator in a wavelet basis. Lower right: Sparsity structure of the inverse operator in a wavelet basis. The values shown are those greater than 10\(^{-12}\) of the maximum in absolute value.
1.4.3.1 A generic form for preconditioning

To see more clearly the benefit of preconditioning by the constant coefficient Green’s function, let us re-organize the Poisson equation as follows. Recalling that \( a(x) \geq \delta > 0 \) for some constant \( \delta \) and all \( x \in \Omega \), we have

\[
-\frac{1}{a}(\nabla \cdot (a\nabla u)) = \frac{f}{a}, \quad \text{(divide by } a) \\
-(\Delta u + \nabla a \cdot \nabla u) = \tilde{f}, \quad \text{(differentiate and distribute } a^{-1}) \\
-(\Delta u + \nabla \ln a \cdot \nabla u) = \tilde{f}, \quad \text{(definition of ln derivative), (1.15)}
\]

where \( \tilde{f} \equiv f/a \). Notice that the operator is now split into a constant piece and variable piece which is “mild” on account of the logarithm, so long as \( \delta \) is not too small.

Letting \( G \) denote the Green’s function corresponding to the constant coefficient problem \( -\Delta u = f \), we apply it on the left hand side to get

\[
-\int_{\Omega} G(x,y)(\Delta u(y) + \nabla \ln a(y) \cdot \nabla u(y))dy = I - \int_{\Omega} G(x,y)\nabla \ln a(y) \cdot \nabla u(y)dy \\
= I - T(x,y),
\]

where \( T(x,y)[u] = \int_{\Omega} G(x,y)\nabla \ln a(y) \cdot \nabla u(y)dy \). Setting \( F(x) = \int_{\Omega} G(x,y)\tilde{f}(y)dy \), the transformed PDE is now,

\[
(I - T(x,y))[u](x) = F(x).
\]

Notice that if \( ||T|| < 1 \), the inverse operator \( (I - T)^{-1} \) could be constructed directly via a Born series by repeated squaring,

\[
(I - T)^{-1} = \sum_{k=0}^{\infty} T^k = \prod_{k=0}^{\infty} (I + T^2)^k.
\]

Even if \( ||T|| \geq 1 \), it is intuitively obvious that the condition number of \( I - T \) should be reasonable so long as the gradients \( \nabla \ln a \) and \( \nabla u \) are also small.
1.4.3.2 A bound for the preconditioned operator

Indeed, preconditioning in this way is known to lead to operators with condition number that depend
on the coefficient via

\[ \kappa(I - T) \leq \frac{\sup_{x \in \Omega} a(x)}{\inf_{x \in \Omega} a(x)}, \]

(see, e.g., [108]). Because the coefficient we consider is never less than some parameter \( \delta > 0 \), we can assume
w.l.o.g. that,

\[ \kappa(I - T) \leq \delta^{-1} \sup_{x \in \Omega} a(x). \] (1.16)

Notice that the condition number no longer depends on the differential part of the operator but merely on
the values of the variable coefficient.

1.5 Numerical results

We consider several numerical examples. In all cases, we precondition with the constant coefficient
Green’s function which is computed analytically using the methods described in Section 1.3.1. For simplicity,
the variable coefficients we use have a sufficiently large constant term so that function \( a^{-1}(x) \) need not be
computed in separated format. Upon preconditioning, we have

\[
\begin{cases}
\mathbb{B}u = G_c f, \\
\mathbb{B} = I + G_c A_v.
\end{cases}
\]

The main computation task is then to construct \( \mathbb{B}^{-1} \) via Schulz iteration.

The errors reported are:

\[
\begin{align*}
\epsilon_1 &= \frac{||\mathbb{B}^{-1}\mathbb{B} - I||_F + ||\mathbb{B}\mathbb{B}^{-1} - I||_F}{2||I||_F}, \\
\epsilon_2 &= \frac{||u - \hat{u}||_F}{||u||_F},
\end{align*}
\]

where \( u \) is a rank one, random tensor with \( u_{ij} \sim \mathcal{N}(0,1) \) for \( i_j = 1, \ldots, M \) and \( j = 1, \ldots, d \). In addition,
the time to apply to a right hand side of separation rank \( r_f = 1, 2, 3 \) is also reported.
Our code is a prototype written in Matlab [2], a proprietary, interpreted language. As such, we present CPU times for the Schulz iteration for larger problems only as an informal reference. We are currently implementing this algorithm in Fortran so that more reliable speed and memory estimates can be made. When times are reported, the following symbols are used to denote the machine type:

\[\dagger = \text{Apple MacBook Pro laptop, 2.66 GHz Intel Core 2 Duo CPU, 4 GB RAM}\]

\[\ddagger = \text{GPU desktop, 2.66 GHz AMD Opteron CPU (\times 16), 132 GB RAM}\]

The code was not explicitly optimized for multi-core computing: the primary reason for using the GPU desktop was increased amount of RAM. The time to apply the operator to the right hand side is reported for the laptop machine.

1.5.1 A coefficient with two terms and mixed boundary conditions \((d = 3)\)

We consider the following Poisson problem on \(D = [0, 1]^3\):

\[
\begin{align*}
  a(x, y, z) &= 1 + \frac{1}{2} \sin(2\pi x) \sin(2\pi y) \sin(2\pi z), \\
  a(0, y, z) &= a(1, y, z), \\
  a(x, 0, z) &= a(x, 1, z), \\
  a(x, y, 0) &= a(x, y, 1) = 0.
\end{align*}
\]

Results are shown in Table 1.3, and the salient features of the Schulz iterates are illustrated in Figure 1.2.

<table>
<thead>
<tr>
<th>(M)</th>
<th>(r_{B-1})</th>
<th>(\mathbf{E}^{-1}), non-zeros</th>
<th>(\epsilon_1)</th>
<th>(\epsilon_2)</th>
<th>(M \setminus r_f)</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>43</td>
<td>(5.0 \times 10^5)</td>
<td>(0.74 \times 10^{-3})</td>
<td>(0.45 \times 10^{-3})</td>
<td>64</td>
<td>1.93</td>
<td>3.93</td>
<td>5.74</td>
</tr>
<tr>
<td>128[^{\dagger}]</td>
<td>33</td>
<td>(1.2 \times 10^5)</td>
<td>(0.20 \times 10^{-3})</td>
<td>(0.21 \times 10^{-3})</td>
<td>128</td>
<td>1.72</td>
<td>3.49</td>
<td>5.20</td>
</tr>
<tr>
<td>256</td>
<td>26</td>
<td>(2.7 \times 10^5)</td>
<td>(0.74 \times 10^{-3})</td>
<td>(0.67 \times 10^{-3})</td>
<td>256</td>
<td>2.02</td>
<td>3.89</td>
<td>5.67</td>
</tr>
<tr>
<td>512</td>
<td>18</td>
<td>(4.0 \times 10^5)</td>
<td>(0.20 \times 10^{-3})</td>
<td>(0.20 \times 10^{-3})</td>
<td>512</td>
<td>2.71</td>
<td>4.71</td>
<td>6.83</td>
</tr>
<tr>
<td>1024[^{\dagger}]</td>
<td>15</td>
<td>(6.0 \times 10^5)</td>
<td>(0.74 \times 10^{-3})</td>
<td>(8.0 \times 10^{-4})</td>
<td>1024</td>
<td>3.17</td>
<td>5.54</td>
<td>7.09</td>
</tr>
</tbody>
</table>

\[^{\dagger}\] 6 min.; \[^{\ddagger}\] 20 min.

Table 1.3: Results for example 1.5.1.
1.5.2 A coefficient with 16 terms in 3D and mixed boundary conditions \((d = 3)\)

We consider the following Poisson problem on \(D = [0, 1]^3\):

\[
\begin{cases}
a(x, y, z) = 15 + \sum_{l=1}^{15} \sin(\beta_1^{(l)} \pi x) \sin(\beta_2^{(l)} \pi y) \sin(\beta_3^{(l)} \pi z), \\
a(0, y, z) = a(1, y, z), \\
a(x, 0, z) = a(x, 1, z), \\
a(x, y, 0) = a(x, y, 1) = 0.
\end{cases}
\]

The frequency coefficient vectors \(\beta^{(l)} = (\beta_1^{(l)}, \beta_2^{(l)}, \beta_3^{(l)}), \ l = 1, \ldots 15\), are various combinations of 1, 2, and 3. For example \(\beta^{(1)} = (1, 1, 1)\) and \(\beta^{(2)} = (1, 3, 2)\). The variable coefficient oscillates between \(\pm 20\%\) of its mean value. Results are shown in Table 1.4.

1.5.3 A coefficient with a sharp peak in 10 dimensions and zero boundary conditions \((d = 10)\)

We consider a coefficient with a sharp peak in the middle of the unit cube in dimension \(d = 10\),

\[
a(x) = 1 - 0.9 \exp(-3 \times 10^3 \cdot (x - 0.5)^2),
\]
This problem may be thought of as “close” to the constant coefficient problem since the peak is expected to be sharply concentrated around $x = 1/2$. This is meant as a proof of concept that the dimension may be easily modified within this framework, but does not correspond to a concrete physical problem that we know of. Results are shown in Table 1.5.

Table 1.4: Results for example 1.5.2.

<table>
<thead>
<tr>
<th>$M$</th>
<th>$r_{B^{-1}}$</th>
<th>$H^{-1}$, non-zeros</th>
<th>$\epsilon_1$</th>
<th>$\epsilon_2$</th>
<th>Time to apply (seconds):</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>66</td>
<td>$5.1 \times 10^4$</td>
<td>$0.43 \times 10^{-3}$</td>
<td>$0.42 \times 10^{-3}$</td>
<td>16</td>
</tr>
<tr>
<td>32</td>
<td>54</td>
<td>$1.7 \times 10^5$</td>
<td>$0.47 \times 10^{-3}$</td>
<td>$0.43 \times 10^{-3}$</td>
<td>32</td>
</tr>
<tr>
<td>64</td>
<td>44</td>
<td>$5.4 \times 10^5$</td>
<td>$0.38 \times 10^{-3}$</td>
<td>$0.45 \times 10^{-3}$</td>
<td>64</td>
</tr>
<tr>
<td>128</td>
<td>34</td>
<td>$1.5 \times 10^6$</td>
<td>$0.37 \times 10^{-3}$</td>
<td>$0.39 \times 10^{-3}$</td>
<td>128</td>
</tr>
</tbody>
</table>

† 17 min.

Table 1.5: Results for example 1.5.3.

<table>
<thead>
<tr>
<th>$M$</th>
<th>$r_{B^{-1}}$</th>
<th>$H^{-1}$, non-zeros</th>
<th>$\epsilon_1$</th>
<th>$\epsilon_2$</th>
<th>Time to apply (seconds):</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>15</td>
<td>$3.8 \times 10^4$</td>
<td>$0.95 \times 10^{-3}$</td>
<td>$0.45 \times 10^{-3}$</td>
<td>16</td>
</tr>
<tr>
<td>32</td>
<td>15</td>
<td>$1.5 \times 10^5$</td>
<td>$0.92 \times 10^{-3}$</td>
<td>$0.90 \times 10^{-3}$</td>
<td>32</td>
</tr>
<tr>
<td>64</td>
<td>15</td>
<td>$5.4 \times 10^5$</td>
<td>$0.89 \times 10^{-3}$</td>
<td>$0.92 \times 10^{-3}$</td>
<td>64</td>
</tr>
<tr>
<td>128</td>
<td>15</td>
<td>$1.6 \times 10^6$</td>
<td>$0.88 \times 10^{-3}$</td>
<td>$0.89 \times 10^{-3}$</td>
<td>128</td>
</tr>
</tbody>
</table>

† 9 min.

1.5.4 Stochastic PDE in one spatial dimension and four stochastic dimensions ($d = 5$)

We consider a stochastic version of the PDE in one physical dimension, with variable coefficient

$$a(x, \omega) = a_0 + \tau \sum_{l=1}^{4} a_l(\omega) e^{-\alpha_l(x-\beta_l)^2}.$$  

Here $\{a_l(\omega)\}_{l=1}^{4}$ are i.i.d. uniform random variables in $U([-1,1])$, and $\{\alpha_l, \beta_l\}_{l=1}^{4}$ are parameters specifying the Gaussian centers and amplitudes. The parameter $\tau$ can be thought of as a noise level since it scales the stochastic terms. One can make the separability of the coefficient explicit by writing,

$$a(x, \omega) = a_0 + \tau \sum_{l=1}^{r} a^{(1)}_l(\omega) a^{(2)}_l(\omega) \cdots a^{(r)}_l(\omega) e^{-\alpha_l(x-\beta_l)^2}, \text{ where}$$

$$a^{(j)}_l = \delta_{jl} a_l,$$
where $\delta_{jl}$ is the Kronecker delta function. As each term in the series has a unique random coefficient, there are $d = r + D$ total dimensions ($r$ stochastic and $D$ physical). For the discretization, we used the second order staggered grid scheme described previously in the spatial dimension. In the stochastic dimension, the distribution is sampled at the Chebychev nodes according to

$$a_{l,k}^{(j)} = -\cos \left( \frac{\pi(k-1)}{M-1} \right), \quad 1 \leq k \leq M.$$ 

We approximated the Green’s function for $r = 4$ and $D = 1$ for a total of $d = 5$ dimensions. The parameters used were: $a_0 = 1$, $\tau = 1/4$, $\alpha_l = 1000 \forall l$, $\beta_l = n/5$ for $n = 1, 2, 3, 4$. Results are shown in Table 1.6.

<table>
<thead>
<tr>
<th>$M$</th>
<th>$r_B^{-1}$</th>
<th>$\mathbb{E}^{-1}$, non-zeros</th>
<th>$\epsilon_1$</th>
<th>$\epsilon_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>36</td>
<td>$5.4 \times 10^5$</td>
<td>$0.15 \times 10^{-4}$</td>
<td>$0.48 \times 10^{-4}$</td>
</tr>
<tr>
<td>256</td>
<td>33</td>
<td>$1.2 \times 10^6$</td>
<td>$0.20 \times 10^{-4}$</td>
<td>$0.55 \times 10^{-4}$</td>
</tr>
<tr>
<td>512</td>
<td>31</td>
<td>$2.0 \times 10^6$</td>
<td>$0.84 \times 10^{-4}$</td>
<td>$0.98 \times 10^{-4}$</td>
</tr>
<tr>
<td>1024</td>
<td>33</td>
<td>$3.6 \times 10^6$</td>
<td>$0.16 \times 10^{-4}$</td>
<td>$0.87 \times 10^{-4}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$M \setminus r_f$</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>0.19</td>
<td>0.26</td>
<td>0.37</td>
</tr>
<tr>
<td>256</td>
<td>0.19</td>
<td>0.28</td>
<td>0.39</td>
</tr>
<tr>
<td>512</td>
<td>0.24</td>
<td>0.32</td>
<td>0.46</td>
</tr>
<tr>
<td>1024</td>
<td>0.35</td>
<td>0.49</td>
<td>0.62</td>
</tr>
</tbody>
</table>

† 94 min.

Table 1.6: Results for example 1.5.4.

### 1.5.5 Stochastic PDE in three spatial dimensions and five stochastic dimensions ($d = 8$)

We consider a stochastic version of the PDE in three physical dimensions, with variable coefficient

$$a(x, \omega) = 1 + \sum_{l=1}^{5} (2l)^{-2}a_l(\omega) \sin(2l\pi x) \sin(2l\pi y) \sin(2l\pi z),$$

where $a_l(\omega)$ are i.i.d. uniform random variables in $U([-1, 1])$. We approximated the Green’s function for $r = 5$ and $D = 3$ for a total of $d = 8$ dimensions. Spatial and stochastic dimensions were discretized as in example 1.5.4. Results are shown in Table 1.7.
Results:

<table>
<thead>
<tr>
<th>$M_{\text{phys}}$</th>
<th>$M_{\text{stock}}$</th>
<th>$r_{B^{-1}}$</th>
<th>$\mathbb{B}^{-1}$, non-zeros</th>
<th>$\epsilon_1$</th>
<th>$\epsilon_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>10</td>
<td>106</td>
<td>$3.3 \times 10^9$</td>
<td>$0.18 \times 10^{-3}$</td>
<td>$0.18 \times 10^{-3}$</td>
</tr>
<tr>
<td>64</td>
<td>10</td>
<td>78</td>
<td>$8.6 \times 10^9$</td>
<td>$0.17 \times 10^{-3}$</td>
<td>$0.33 \times 10^{-3}$</td>
</tr>
<tr>
<td>128</td>
<td>10</td>
<td>60</td>
<td>$1.9 \times 10^9$</td>
<td>$0.16 \times 10^{-3}$</td>
<td>$0.27 \times 10^{-3}$</td>
</tr>
<tr>
<td>$256^*$</td>
<td>10</td>
<td>45</td>
<td>$3.2 \times 10^9$</td>
<td>$0.19 \times 10^{-3}$</td>
<td>$0.21 \times 10^{-3}$</td>
</tr>
<tr>
<td>$512^*$</td>
<td>10</td>
<td>33</td>
<td>$5.3 \times 10^9$</td>
<td>$0.15 \times 10^{-3}$</td>
<td>$0.32 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

*† 48 min., ‡ 90 min.

Table 1.7: Results for example 1.5.5.

1.5.6 Discussion

In our experiments, we observed that the separation rank of the inverse operator $(I - T)^{-1}$ would often decrease with increasing problem size. This is somewhat unexpected since, theoretically, the rank is expected to increase mildly with respect to $M$. We attribute this to the fact that, for small problem sizes, the discretization error dominates and the separated representation is effectively capturing artifacts (“noise”). By around $M = 1024$ (feasible for example 1.5.1), the discretization error is on the same order as the desired accuracy. Thus, for $M > 1024$, we would expect the ranks to stabilize and remain roughly constant.

Accuracies of 3-5 digits were obtainable in all cases. As illustrated in Figure 1.2, extremely low separation rank can be maintained for most of the Schulz iteration, only to increase as the quadratic convergence kicks in and the error begins to drop quickly. Although not shown, similar trends hold for all of the examples. It is apparent from the figure that an additional iteration or two will further increase the accuracy (as well as the separation rank) of the inverse operator. The primary reason this was not done was time (i.e., time constraints on the author).
Another feature common to all of the experiments is the relatively mild increase in the number of non-zeros in the inverse operators, with respect to problem size $M$. Note that a “full” tensor operator of the kind we consider has $M^{2d}$ entries (each of the $d$ dimensions contains an $M \times M$ matrix). So, formally, doubling $M$ leads to a factor of $2^{2d}$ increase in the total number of numbers in the full operator. In our experiments, the number of non-zeros increases by a factor of $2 - 4$ which, in comparison to $2^{2d}$, is extremely mild (recall, $3 \leq d \leq 10$ in all cases).

Finally, as expected by a simple operation count, the time to apply the Green’s function to the right hand side scales linearly in $r_f$ (the separation rank of $f$) and only mildly on $M$. There are two reasons for the mild dependence on $M$. On the one hand, the issue of the discretization error mentioned above typically leads to lower separation rank of the operators as $M$ increases, thus offsetting the growth in problem size. On the other hand, regardless of the discretization error, we expect mild growth ($O(M \log M)$) on account of the sparsity Green’s function in the standard form.
Chapter 2

Fast Randomized Algorithms for Canonical Tensor Decompositions
2.1 Introduction

Let $U \in \mathbb{R}^{M_1 \times \cdots \times M_d}$ be a $d$-dimensional array that admits a canonical tensor decomposition (CTD) of separation rank $r$,

\begin{align}
U_{i_1, \ldots, i_d} &= \sum_{l=1}^{r} \sigma_l \prod_{j=1}^{d} u_{i_j}^{(l)} \quad \text{or} \quad (2.1) \\
U &= \sum_{l=1}^{r} \sigma_l \bigotimes_{j=1}^{d} u_{j}^{(l)}, \quad (2.2)
\end{align}

where the weights $\sigma_l$ are chosen so that $\|u_{j}^{(l)}\|_2 = 1$ for all $j, l$. As in Chapter 1, we will use calligraphic font to denote a generic tensor, “blackbord” font (with the exception of $\mathbb{R}$, which is reserved for the set of real numbers) to denote an operator, and lowercase bold to denote pointwise-defined functions.

In numerical applications, it is possible to encounter canonical tensors having extremely high nominal separation rank. For example, suppose $U$ represents a multi-dimensional operator and that the self-adjoint product $U^* U$ is sought. Clearly, if $U$ has rank $r$, then $U^* U$ will have rank $r^2$. For even moderate separation ranks, it may become very expensive to compute basic quantities such as Frobenius norm of such an object.

To see this, recall that the Frobenius norm for an arbitrary tensor is defined via,

\[ \|U\|_F^2 = \langle U, U \rangle = \sum_{l=1}^{r} \sum_{m=1}^{r} \sigma_l \sigma_m \prod_{j=1}^{d} \langle u_{i_j}^{(l)}, u_{i_j}^{(m)} \rangle \quad (2.3) \]

To simplify expressions, and without loss of generality, let us assume that $M_1 = M_2 = \cdots = M_d = M$. Computation of $\|U\|_F$ then requires $O(d \cdot M \cdot r^2)$ operations. The cost of computing $\|U^* U\|_F$, on the other hand, is $O(d \cdot M \cdot r^4)$ and hence quartic in the original separation rank. Similar arguments show that the use of standard rank reduction algorithms, such as alternating least squares (ALS), also become problematic for seemingly reasonable values of $r$.

In many applications, however, large separation rank is an artifact of the algebraic constraints of the canonical decomposition: to multiply two such objects requires $r^2$ terms in the result because we cannot easily “collect like terms” in the process. On the other hand, for a given accuracy the true separation rank may be much lower than its nominal value. It may even be redundant in the sense that many of the terms are (approximately) linearly dependent. This is exactly the situation we would like to address.
2.2 Analytical framework

2.2.1 Interpolative decomposition of a matrix

The analytical framework from our approach comes from [37, 148, 100], in which the authors describe an algorithm for the interpolative decomposition (ID) of low rank matrices. The relation to the tensor case will be made clear below. In the meantime, we summarize the key results.

**Theorem 2.** (Cheng, Gimbutas, Martinsson and Rokhlin) Suppose that $A$ is an $m \times n$ matrix and let $k$ be such that $1 \leq k \leq \min(m, n)$. Then there exists a interpolative decomposition,

$$A = A_{CS}[I|T_{CS}]P_R^* + X,$$

where the matrix $A_{CS}$ (the “column skeleton”) contains $k$ columns of $A$, $I$ is the $k \times k$ identity matrix, and $T_{CS} \in \mathbb{R}^{k \times (n-k)}$ and $P_R \in \mathbb{R}^{n \times n}$ are matrices derived from the QR factorization of $A$. Furthermore, we have that

$$||T_{CS}||_F \leq \sqrt{k(n-k)},$$

$$||X||_2 \leq \sigma_{k+1}(A)\sqrt{1 + k(\min(m, n) - k)},$$

where $\sigma_{k+1}(A)$ is the $(k+1)$th singular value of $A$.

**Remark.** The numerical apparatus underlying Theorem 2 involves computing the pivoted, re-orthogonalized QR factorization of $A$ until the column space has been exhausted to within an accuracy $\epsilon$, so that $A$ can be expressed

$$AP_R = Q[R_{11}|R_{12}],$$

where $P_R \in \mathbb{R}^{n \times n}$ is a permutation matrix, $Q \in \mathbb{R}^{m \times k}$ has orthonormal columns, $R_{11} \in \mathbb{R}^{k \times k}$ is upper triangular, and $R_{12} \in \mathbb{R}^{k \times (n-k)}$ (see [60], Chapter 5.2 for an overview of QR algorithms). Indeed, the matrix $P_R$ in the theorem is exactly this permutation matrix, and $T_{CS} = R_{11}^{-1}R_{12}$. When $R_{11}$ is ill-conditioned, $T_{CS}$ may be selected from the many possible solutions by minimizing $||T_{CS}||_F.$
2.2.2 Randomized algorithm for computing the ID of a low-rank matrix

The deterministic calculation of the decomposition in Theorem 2 is known to be expensive. However, when \( A \) is low-rank it may be constructed more efficiently using a randomized algorithm as described in [148, 100]. For convenience, define

\[
P = [I|T_{CS}]P_R^*.
\]

(2.4)

Then \( A_{CS} \) and \( P \) may be constructed efficiently, albeit satisfying somewhat weaker conditions than those in Theorem 2, such that

\[
||A_{CS}P - A||_2 \lesssim \sigma_{k+1}(A).
\]

(2.5)

An algorithm for doing so is summarized as follows:

(1) Compute a matrix \( Y = RA \), where \( R \in \mathbb{R}^{l \times m} \) is a matrix of independent, identically distributed (i.i.d.) Gaussian random variables (r.v.’s) with zero mean and unit variance.

(2) Using Theorem 2, form the matrix \( Z \) as a collection of \( k \) appropriately chosen columns of \( Y \). In addition, construct the matrix \( P \) in (2.4) so that \( ||ZP - Y||_2 \leq \tau_{k+1} \), where \( \tau_{k+1} \) is the \((k+1)\)st largest singular value of \( Y \).

(3) Collect the columns of \( A \) corresponding to the \( k \) columns of \( Y \) into the matrix \( A_{CS} \). This is the column skeleton of \( A \).

If we let \( c_A \) denote the cost of applying \( A^* \) to a vector, then the randomized procedure costs at most \( l \cdot c_A + O(k \cdot m + k \cdot l \cdot n) \) (see [148, 100] for details).

2.3 Application to tensor norm and rank reduction

In this section, we describe how some important operations on canonical tensors may be cast in terms of inner products of its terms using a novel representation. We sketch an algorithm for rank reduction that, for certain problems, may be far more efficient than using conventional rank reduction techniques. Finally, we provide error estimates for the reduction essential to any practical algorithm involving such objects.
2.3.1 A useful matrix representation for CTD’s

Here we introduce a somewhat abstract representation of the CTD that makes the connection to the randomized algorithms more natural. We define a canonical matrix $\mathbf{U}$ of a tensor $\mathcal{U}$ as,

$$
\mathbf{U} = \begin{pmatrix}
\sigma_1 \mathcal{U}^{(1)} & \sigma_2 \mathcal{U}^{(2)} & \cdots & \sigma_r \mathcal{U}^{(r)} \\
\vdots & \vdots & \ddots & \vdots \\
\end{pmatrix},
$$

such that:

- $\mathcal{U}^{(l)}$ has separation rank one,
- $\|\mathcal{U}^{(l)}\|_F = 1$, and
- $\mathcal{U} = \sum_{l=1}^r \sigma_l \mathcal{U}^{(l)}$.

Notationally, $\mathbf{U}$ is distinguished from a standard matrix by a bold serif font and curved brackets.

The defining difference between the canonical matrix and a standard matrix is that the columns of $\mathbf{U}$ are multilinear objects, with $d \cdot M$ degrees of freedom defined by the CTD parameters. While in principle we could consider each column to be an $M^d$-vector (by “vectorizing” each term in the separated representation), doing so is practically intractable as well as misleading, owing to the small number of parameters that fully describe each column.

Instead, we sidestep this issue by restricting the operations involving $\mathbf{U}$ to the following:

- Transposition, as in:

$$
\mathbf{U}^* = \begin{pmatrix}
\sigma_1 \mathcal{U}^{(1)} & \vdots & \sigma_r \mathcal{U}^{(r)} \\
\vdots & \ddots & \vdots \\
\end{pmatrix}
$$

- Column-wise inner products $\langle \mathcal{U}^{(l)}, \mathcal{U}^{(m)} \rangle$, defined in the standard way.

- Right-multiplication by a vector or matrix. For example, we have $\mathcal{U} = \mathbf{U} \mathbf{1}$, where $\mathbf{1}$ is an $r$-vector of ones.
As we will describe, these three operations are sufficient for our purposes, and an explicit definition of the rows of $\mathbf{U}$ is never needed.

Operationally, the canonical matrix representation is useful for two reasons. First, it leads to simplified core expressions. Let $\mathbf{G}(\mathcal{U})$ denote the term-wise Gram matrix of tensor $\mathcal{U}$,

$$G_{lm}(\mathcal{U}) = \langle U^{(l)}, U^{(m)} \rangle,$$

or equivalently,

$$\mathbf{G}(\mathcal{U}) = \mathbf{U}^* \mathbf{U}.$$

Similarly, define the term-wise Gram matrix between two distinct tensors as

$$\mathbf{G}(\mathcal{U}, \mathcal{V}) = \mathbf{U}^* \mathbf{V}.$$

Comparing with equation (2.3), we note that the Frobenius norm has a simple expression in terms of the canonical matrix,

$$\|\mathcal{U}\|_F^2 = \sum_{l=1}^{r} \sum_{m=1}^{r} G_{lm}(\mathcal{U}) = 1^* \mathbf{U}^* \mathbf{U} 1,$$

where $1$ is a ones vector in $\mathbb{R}^r$.

The second reason for using the canonical matrix format is that we may consider computing an ID of tensor $\mathcal{U}$ via,

$$\mathcal{U} \approx \mathcal{U}_{\text{CS}} \mathbf{P},$$

where $\mathbf{P}$ is a mixing matrix like that in (2.4). We will describe this in detail below, but for now note that the algorithms in Section 2.2 may all be cast in terms of transposition and inner products of the low rank matrix whose decomposition is sought. Hence, operationally, the canonical matrix format is identical to the standard matrix format in terms of the ID algorithm.

We now are in a position to address two important questions. If the nominal separation rank of a canonical tensor is large, but we expect that it is much smaller for a given accuracy, how can this information be leveraged to:
• efficiently compute the Frobenius norm?

• efficiently compute a much lower rank - if not necessarily optimal - approximation?

We turn to these questions next.

2.3.2 An efficient algorithm for the Frobenius norm

For concreteness, let us consider the case of the self-adjoint operator mentioned in the introduction. We begin with an object of rank \( r \) that, upon multiplication with its adjoint, becomes an object of nominal rank \( r^2 \). As was mentioned above, direct calculation of the Frobenius norm of the result has complexity \( O(d \cdot M \cdot r^4) \).

However, let us suppose that \( \mathcal{U} \) is low rank in the following sense: its canonical matrix representation admits an ID with \( k \ll r^2 \) “skeleton” terms,

\[
\mathbf{U} = \mathbf{U}_{CS} \mathbf{P} + \mathbf{X},
\]

where \( \|\mathbf{X}\| \) is “small”. Examining the randomized algorithm for computing a matrix ID in Section 2.2.2, we see that all of the operations needed in the matrix setting are extensible to the tensor setting. Therefore, to efficiently compute the matrices \( \mathbf{U}_{CS} \) and \( \mathbf{P} \), we propose the following algorithm:

1. Form the Gram matrix \( \mathbf{Y} = \mathbf{G}(\mathcal{R}, \mathcal{U}) = \mathbf{R}^* \mathbf{U}, i = 1, \ldots, \ell \) and \( j = 1, \ldots, r \). Here \( \mathcal{R} \) satisfies,

\[
\mathcal{R} = \sum_{m=1}^{\ell} \bigotimes_{j=1}^{d} \mathbf{r}_{ij}^{(m)},
\]

\[
\hat{\mathbf{r}}_{ij}^{(m)} \sim \mathcal{N}(0, 1),
\]

\[
\mathbf{r}_{ij}^{(m)} = \frac{\mathbf{r}_{ij}^{(m)}}{\|\mathbf{r}_{ij}^{(m)}\|_2}.
\]

2. Using the algorithm described in the previous section, identify the \( k \) terms of \( \mathcal{U} \) and compute the corresponding matrix, \( \mathbf{P} \).
In the process of computing the above quantities, we are required to compute the QR factorization of $Y$ such that $Y P_Y = QR$, with $P_Y$ a permutation matrix. (This will be useful for the rank reduction technique described below).

With the ID (2.6) in hand, we are now in a position to compute $||\mathcal{U}||^2_F$. We have simply

$$||\mathcal{U}||^2_F = 1^* U^* U 1$$

$$= 1^* P^* U_{CS} U_{CS} P 1$$

$$= 1^* P^* G(\mathcal{U}_{CS}) P 1,$$

where $1 \in \mathbb{R}^r$. The essential point here is that the full $r \times r$ Gram matrix $G(\mathcal{U})$ need not be formed. Instead, only only has to form the $k \times k$ “skeleton” Gram matrix, followed by operations that cost at most $O(k \cdot r)$.

Next we estimate the total cost of computing the Frobenius norm using the tensor ID. First, note that in the cost estimate of computing the standard matrix ID, the constant $c_A$ represents the cost of applying the matrix $A^*$ to a vector. In the tensor setting, this is the cost of taking the inner product of $\mathcal{U}$ with a random, rank-one tensor. Hence we replace $c_A$ with $O(r \cdot d \cdot M)$. The remainder of the steps, whose cost is straightforward to estimate, are tabulated in Table 2.1.

While the total cost of the above algorithm does not have a simple expression, let us hold all but the rank variables constant to compare the cost in terms of rank. We see that to leading order it is $O(r^2(k + l) + k \cdot l \cdot r^2 \log(r))$, versus $O(r^4)$ for the full rank-$r^2$ tensor. Note that the $O(k \cdot l \cdot r^2 \log(r))$ term, although it will reach $O(r^4)$ if $k \cdot l = O(r^2 \log^{-1}(r))$, is $O(1)$ in both $M$ and $d$. The standard Frobenius norm, in contrast, is linear in both of these variables. Hence, for large, high-dimensional tensors, the norm may be computed at a significant savings using the above approach.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Construct the ID</td>
<td>$O(d \cdot M \cdot r \cdot (k + l) + k \cdot l \cdot r \log(r))$</td>
</tr>
<tr>
<td>Construct the Gram matrix, $G(\mathcal{U})$</td>
<td>$O(d \cdot M \cdot k^2)$</td>
</tr>
<tr>
<td>Matrix-vector multiplication, $p = P1$</td>
<td>$O(k \cdot r)$</td>
</tr>
<tr>
<td>Vector-matrix-vector multiplication, $p^* G(\mathcal{U}) p$</td>
<td>$O(k^2)$</td>
</tr>
</tbody>
</table>

Table 2.1: Operation counts for randomized ID of a canonical tensor.
2.3.3 A tensor ID: efficient, non-optimal rank reduction

In the process of describing an efficient way to compute $\|U\|_F$, we have also laid the groundwork (and done practically all of the computational work) for computing a non-optimal - but possibly much lower-rank - approximation of $\mathcal{U}$. The basic idea is that if the terms collected in the skeleton matrix $U_{CS}$ (i.e., a subset of the terms in the separated representation of $\mathcal{U}$) are sufficient to represent $\mathbf{U}$, perhaps they will also be useful for representing $\mathcal{U}$ as a tensor.

In particular, let us consider finding coefficients $\{\alpha_i\}_{i=1}^k$ such that

$$
\mathcal{U} = \sum_{i=1}^k \alpha_i \sigma_i U^{(l_i)},
$$

(2.8)

where the indices $l_i$ correspond to the $k$ skeleton terms in the ID (2.6). Letting $P_{ij}$ be the elements of the matrix $P$ in (2.6), we re-express that equation as,

$$
\mathcal{U} = \sum_{i=1}^k \sum_{j=1}^r P_{ij} \sigma_i U^{(l_i)}.
$$

Indeed, setting

$$
\alpha_i = \sigma_i \sum_{j=1}^r P_{ij},
$$

(2.9)

we get the desired result: the coefficient is obtained by simply summing the rows of matrix $P$ and scaling by the original weight.

2.3.4 An error estimate for the tensor ID

Here we derive a simple error bound for non-optimal rank reduction via the randomized tensor ID.

**Proposition 1.** Let $\mathbf{U} = [\sigma_1 \mathcal{U}^{(1)}, \sigma_2 \mathcal{U}^{(2)}, \ldots, \sigma_r \mathcal{U}^{(r)}]$ be a canonical matrix containing the rank-one terms of tensor $\mathcal{U}$ in its columns. Suppose that $\|\mathbf{U} - U_{CS}P\|_2 \leq \epsilon$ where $U_{CS}$ is the column skeleton of $\mathbf{U}$ and $P$ is the matrix defined in (2.4). Then, letting $\mathcal{U}_k$ denote the canonical representation of $U_{CS}P$, $\mathcal{U}_k$ satisfies

$$
\|\mathcal{U} - \mathcal{U}_k\|_F^2 \leq r\epsilon.
$$
Proof. First, note that we can bound the Frobenius norm of $\mathcal{U}$ via,

$$||\mathcal{U}||_F^2 = \sum_{l=1}^{r} \sum_{m=1}^{r} \sigma_l \sigma_m \prod_{j=1}^{d} \langle \mathbf{u}_j^{(l)}, \mathbf{u}_j^{(m)} \rangle \leq \sum_{l=1}^{r} \sum_{m=1}^{r} \sigma_l \sigma_m = (\sum_{l=1}^{r} \sigma_l)^2,$$

which follows from the Cauchy-Schwartz inequality and the fact that $||\mathbf{u}_j^{(l)}|| = 1$ for $j = 1, \ldots, d$ and $l = 1, \ldots, r$. The norm of the matrix $\mathbf{U}$ in the sense of (2.6) may be expressed,

$$||\mathbf{U}||_F^2 = \text{trace}(\mathbf{U}^* \mathbf{U}) = \sum_{l=1}^{r} \sigma_l^2,$$

since the trace of $\mathbf{U}^* \mathbf{U}$ contains only inner products of the $r$ terms with themselves, and $\langle \sigma_l \mathbf{u}_j^{(l)}, \sigma_l \mathbf{u}_j^{(l)} \rangle = \sigma_l^2$. Recall that any sequence of real numbers $\{a_l\}_{l=1}^{r}$ satisfies $(\sum_{l=1}^{r} a_l)^2 \leq r \sum_{l=1}^{r} a_l^2$. In addition, the Frobenius and $\ell_2$ norm equivalence produces $||\mathbf{U}||_F^2 \leq r ||\mathbf{U}||_2^2$, from which we estimate

$$||\mathcal{U}||_F^2 \leq r ||\mathbf{U}||_2^2 \leq r^2 ||\mathbf{U}||_F^2.$$

Finally, for the error in approximating $\mathcal{U}$ by its ID,

$$||\mathcal{U} - \mathcal{U}_k||_F \leq r ||\mathbf{U} - \mathbf{U}_{\text{CS}} \mathbf{P}||_2 \leq r \epsilon.$$

(2.11)

Remark. The upper bound appears to be theoretically tight but, in practice, we expect it to be somewhat pessimistic. To see why, consider how to achieve the upper bound in (2.10). In order for all inner products to equal one, the tensor must be highly degenerate, with all factors identical to one other. In structured problems, many of these inner products will be small, leading to a much smaller upper bound.

2.3.5 An a posteriori error estimate

A key challenge in using the randomized approach for rank reduction is that, in general, one does not have access to the singular values of the canonical matrix $\mathbf{U}$ that are implicit in the bound (2.11) via
Theorem 2. Indeed, we have been intentionally vague about the exact entries of this matrix and so the singular values are not even well-defined. So the question remains:

- “Given a rank-\(k\) approximation \(U_k\) to \(U\) found via randomized rank reduction, how do we estimate the error in the approximation without explicitly computing \(\|U - U_k\|_F^2\), i.e., with less than \(O(d \cdot M \cdot r^2)\) operations?”

Here we describe an algorithm to approximate the error efficiently. We do so for a tensor with dimension \(d = 3\), although the idea generalizes easily to higher dimensions.

Let \(A = \sum_{l=1}^{r} \sigma_l a^{(l)} \otimes b^{(l)} \otimes c^{(l)}\) be an arbitrary canonical tensor of rank \(r\). Suppose we wish to find an optimal rank-one approximation \(X = \lambda x \otimes y \otimes z\) that minimizes \(\|A - X\|_F^2\), where \(\|x\| = \|y\| = \|z\| = 1\).

Using Lagrange multipliers, one can show that critical points are solutions to the system of equations

\[
\begin{align*}
\sum_{l=1}^{r} \sigma_l \langle b^{(l)}, y \rangle \langle c^{(l)}, z \rangle a^{(l)} &= \lambda x, \\
\sum_{l=1}^{r} \sigma_l \langle a^{(l)}, x \rangle \langle c^{(l)}, z \rangle b^{(l)} &= \lambda y, \\
\sum_{l=1}^{r} \sigma_l \langle a^{(l)}, x \rangle \langle b^{(l)}, y \rangle c^{(l)} &= \lambda z, \\
\sum_{l=1}^{r} \sigma_l \langle a^{(l)}, x \rangle \langle b^{(l)}, y \rangle \langle c^{(l)}, z \rangle &= \lambda,
\end{align*}
\]

(2.12)

where \(\lambda \geq 0\) may be though of as a singular value of \(A\) (see [152], and Chapter 1.2.4.2 of this thesis). Solving such a system can be shown to be equivalent to solving the dual problem,

\[
\max_{x,y,z} \lambda = \max_{x,y,z} \sum_{l=1}^{r} \sigma_l \langle a^{(l)}, x \rangle \langle b^{(l)}, y \rangle \langle c^{(l)}, z \rangle,
\]

(2.13)

where \(\|x\| = \|y\| = \|z\| = 1\).

One way to compute \(\lambda, X\) satisfying the equations (2.12) is to use a Gauss-Seidel-like iteration. Specifically, initialize \(X\) randomly to have norm one. Then compute \(x\) via the first line in (2.12), and update \(\lambda \leftarrow \|x\|, x \leftarrow x/\lambda\). Do the same for \(y\) and \(z\), and repeat until \(\lambda\) does not change by much.

Suppose we have found \(X\) maximizing (2.13). Then \(\lambda\) may be used to bound the Frobenius norm of \(A\).
Lemma. Suppose $\lambda$ and $x, y, z$ are given satisfying the maximization problem (2.13). Then

$$||A||_F \leq \left( \lambda \sum_{l=1}^{r} \sigma_l \right)^{1/2}. \quad (2.14)$$

Proof. The proof follows in a straightforward way from re-arranging the expression for the Frobenius norm. Because the weights $\sigma_l$ are all positive, we have

$$||A||_F^2 = \sum_{l=1}^{r} \sum_{m=1}^{r} \sigma_l \sigma_m \left\langle a^{(l)}, a^{(m)} \right\rangle \left\langle b^{(l)}, b^{(m)} \right\rangle \left\langle c^{(l)}, c^{(m)} \right\rangle,$$

$$\leq \sum_{m=1}^{r} \sigma_m \left| \sum_{l=1}^{r} \sigma_l \left\langle a^{(l)}, a^{(m)} \right\rangle \left\langle b^{(l)}, b^{(m)} \right\rangle \left\langle c^{(l)}, c^{(m)} \right\rangle \right|.$$ \quad (2.15)

Now, for a given $m$, the sum within the absolute value must always be less than or equal to $\lambda$ since $||a^{(m)}|| = ||b^{(m)}|| = ||c^{(m)}|| = 1$ and, by assumption, $\lambda$ is an upper bound for all such expressions since it satisfies (2.13). Therefore,

$$||A||_F^2 \leq \sum_{m=1}^{r} \sigma_m \left| \sum_{l=1}^{r} \sigma_l \left\langle a^{(l)}, x \right\rangle \left\langle b^{(l)}, y \right\rangle \left\langle c^{(l)}, z \right\rangle \right|,$$

$$\leq \lambda \sum_{m=1}^{r} \sigma_m.$$

Remark. A simpler but looser bound that explicitly emphasizes the dependence on separation rank is given by $||A||_F \leq \sqrt{\lambda r \sigma_{\text{max}}}$, where $\sigma = \max_j \sigma_j$.

Remark. The fixed point iteration is only guaranteed to converge to a local extremum. However, it is possible to ensure that the bound is clean regardless of the true value of $\lambda$. In particular, one can initialize the fixed point iteration with $X_0 = a^{(l)} \otimes b^{(l)} \otimes c^{(l)}$ for $l = 1, \ldots, r$. Because the iteration is monotone increasing with respect to $\lambda$, the local maximum $\lambda_l$ will provide a valid upper bound for the $l$-th term in the inequality (2.15). One can then select $\lambda = \max \lambda_l$. In practice, we have observed that selecting $X_0$ to be the term with the largest weight is usually sufficient to obtain a valid value of $\lambda$. 
Setting $A = U - U_k$, the above scheme provides an efficient way to estimate the error in the tensor ID. Note that direct calculation of the Frobenius norm costs $O(d \cdot M \cdot r^2)$, while the above method requires only $O(d \cdot M \cdot n \cdot r)$ where $n$ is the number of fixed point iterations. Hence the savings is significant when $n \ll r$.

2.4 Numerical results

We present results for several numerical experiments illustrating the upper bound (2.14) as well as the randomized rank reduction procedure. For the bound, we will consider both random and structured tensors corresponding to the BVP in Chapter 1. For the randomized rank reduction, we will consider the Newton-Schulz iteration used to compute the inverse operator of the preconditioned problem.

For reference, we recapitulate the problem of interest from Chapter 1. We consider the variable coefficient Poisson equation in dimension $d$,

\[
\begin{cases}
-\nabla (a(x) \nabla u(x)) = f(x), & x \in D, \\
u(x) = g(x), & x \in \partial D,
\end{cases}
\]  
(2.16)

where $a(x) \geq \delta > 0$. We denote by $\mathbb{A} = \mathbb{A}_c + \mathbb{A}_v$ the discretized operator corresponding to $-(\Delta + \nabla \ln a(x) \cdot \nabla)$, with $\mathbb{A}_c \approx -\Delta$ and $\mathbb{A}_v \approx -\nabla \ln a(x) \cdot \nabla$. We precondition with the constant coefficient Green’s function, $G_c$, leading to a system of equations of the form

\[ \mathbb{B}u = G_c f/a, \]

where $\mathbb{B} = I + G_c A_v$.

To illustrate the bound (2.14), we consider tensors of two distinct types:

1. Random tensors $u$ of various ranks $r$, number of dimensions $d$, and dimension sizes $M$, where $[u_j^{(i)}] \sim \mathcal{N}(0,1)$ before normalization. We also consider highly degenerate tensors of this type.

2. Operators of the BVP (2.16). Specifically, we let $D = [0,1]^3$ with $u = 0$ on $\partial D$ with

\[ a(x) = 1 + \frac{1}{2} \sin(2\pi x) \sin(2\pi y) \sin(2\pi z), \]

and consider the operators $\mathbb{A}$, $G_c$, $\mathbb{B}$ and $\mathbb{B}^{-1}$ for various problems sizes.
2.4.1 Illustration of the error estimate.

2.4.1.1 Gaussian random tensors

We first constructed 1000 Gaussian random tensors with \(d = 3\), \(N = 16\) and \(r = 4, 8, 16, 32\). We computed the Frobenius norm of these tensors in the standard way as well as the upper bound (2.14). The results are shown in Figure 2.1, with bounds sorted in decreasing order. In addition, we considered 1000 Gaussian random tensors in dimensions \(d = 3, 5, 7\) with \(r = 1000\) and \(M = 100\) (Figure 2.2). In both of these cases, the upper bound could be comfortably obtained by choosing the initial guess for the fixed point iteration as the term with the largest weight \(\sigma_l = \sigma_{\text{max}}\).

![Figure 2.1: True and estimated Frobenius norm for random tensors with \(d = 3\), \(M = 16\) and \(r = 4, 8, 16, 32\).](image)

We also wanted to consider the case of degenerate tensors, in which many of the factors are nearly collinear. As such, we constructed the following example:

\[
u = \sum_{l=1}^{10} \sigma_l^u (u_1 + \epsilon_1^{(l)}) \otimes (u_2 + \epsilon_2^{(l)}) \otimes (u_3 + \epsilon_3^{(l)}),
\]

where \(\epsilon_j^{(l)}\) are small normal perturbations in each dimension \(j\). Note the lack of \(l\)-superscript on the vectors \(u_j\), indicating that they are the same for all terms in the series. In particular, \([u_j]_i \sim \mathcal{N}(0, 1)\) and \([\epsilon_j]_i \sim \mathcal{N}(0, 10^{-3})\) before normalization. The bound was computed for 1000 realizations with \(M = 20\). In this case, the bound is extremely close to the true value with only around 3-4 digits relative error (not shown). Again,
this could be achieved by running only one fixed point iteration to obtain $\lambda$, with the initial guess chosen as the term with the largest weight.

### 2.4.1.2 Structured tensors of the BVP

Next we consider the discretized operators $A, G_c, B$ and $B^{-1}$ of the BVP (2.16) for $M = 64, 128, 256,$ and 512 (Table 2.2). We ran the fixed point iteration until the estimate of $\lambda$ changed by less than three digits (usually under 20 iterations). Because the cost of the fixed point method is $O(d \cdot M \cdot r \cdot n)$, where $n$ is the number of iterations needed, versus $O(d \cdot M \cdot r^2)$ for the Frobenius norm, we expect the iterative method to provide a “fast” estimate when $n \ll r$. Indeed, this is exactly what happened in the cases when the bound time was significantly less than the Frobenius norm time. The difference is expected to be even more dramatic for large separation ranks (see next section).
2.4.2 Randomized rank reduction of Schulz iterates

We illustrate the rank reduction procedure by considering the estimation of $B^{-1}$ (the inverse operator of the BVP) via Schulz iteration,

$$X_{n+1} = 2X_n - X_nBX_n,$$

$$X_1 = \alpha B^*,$$

with $0 < \alpha < 1$. Denote the separation rank of $B$ by $r_B$ and that of $X_n$ by $r_n$. The iteration will clearly cause the nominal rank of the approximate inverse to grow rapidly, since $r_{n+1} = r_n + r_n^2 r_B$. Even by the second iteration - if no rank reduction is performed - the rank of the approximate inverse is already greater than $r_B^3$. For even moderate values of $r_B$ and problem size $M$, even computing to norm of such an object is prohibitive.

To illustrate the efficacy of the randomized technique, we examine both deterministic and randomized reduction of the operator $B^*B$. Such quantities are formed throughout the Schulz iteration (indeed, with the initial iterate set to $X_1 = \alpha B^*$, this is exactly the form of the first matrix-matrix multiplication).

In what we call the “deterministic” approach, the product $B^*B$ is formed and reduced via guided ALS (gALS). As the desired accuracy is relative accuracy, we are required to compute the Frobenius norm of the full tensor up front, and so we count this cost as part of the deterministic reduction procedure (it is only computed once).

For the “deterministic plus randomized” approach, we used an upper bound on the Frobenius norm as estimated by the fixed point iteration. This typically took between 3 and 10 iterations ($n \ll r$) until $\lambda$ changed by less than five digits in a relative sense. This upper bound is used to estimate the tolerance for randomized rank reduction (RRR) step, in which the tensor ID is computed for $k = 25j$, $j = 1, 2, 3, \ldots$ until the tolerance is met. Instead of computing the true Frobenius error for each $k$, we again used fixed point iteration to estimate an upper bound on the error. Once the RRR step was performed, the separation rank was now within reasonable limits for use with gALS, and so a final reduction step was performed to further optimize the rank.

The results are summarized in Table 2.3. Relatively small problem sizes, $M = 32, 64$ and 128 were
considered so that the true errors could be computed and compared (recall that each dimension contains $M \times M$ sparse matrices). The left column shows the times and computed values for the completely deterministic approach using gALS. The left column shows analogous quantities using RRR + gALS reduction step. For $M = 128$, the initial separation rank of $r_Y = 1600$ was too large for gALS to be performed given the amount of available RAM ($\sim 4$ GB). The randomized approach, however, was able to handle this problem comfortably.
<table>
<thead>
<tr>
<th>Operator</th>
<th>128 ($r = 6$)</th>
<th>256 ($r = 6$)</th>
<th>512 ($r = 6$)</th>
<th>1024 ($r = 6$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(\lambda \sum_{l=1}^r \sigma_l)^{1/2}$</td>
<td>$1.08 \times 10^4$</td>
<td>$3.06 \times 10^4$</td>
<td>$8.66 \times 10^4$</td>
<td>$2.45 \times 10^5$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>A</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$Time_{b\text{ound}}$ (sec.)</td>
<td>0.068</td>
<td>0.169</td>
<td>0.220</td>
<td>1.009</td>
</tr>
<tr>
<td>$Time_{\text{true}}$ (sec.)</td>
<td>0.031</td>
<td>0.115</td>
<td>0.148</td>
<td>0.837</td>
</tr>
<tr>
<td>$G_c$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(\lambda \sum_{l=1}^r \sigma_l)^{1/2}$</td>
<td>$1.57 \times 10^4$</td>
<td>$6.49 \times 10^4$</td>
<td>$2.61 \times 10^4$</td>
<td>$1.01 \times 10^5$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>G_c</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$Time_{b\text{ound}}$ (sec.)</td>
<td>0.321</td>
<td>0.489</td>
<td>1.300</td>
<td>2.264</td>
</tr>
<tr>
<td>$Time_{\text{true}}$ (sec.)</td>
<td>0.601</td>
<td>2.218</td>
<td>3.338</td>
<td>2.631</td>
</tr>
<tr>
<td>$B$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(\lambda \sum_{l=1}^r \sigma_l)^{1/2}$</td>
<td>$1.58 \times 10^4$</td>
<td>$4.46 \times 10^4$</td>
<td>$1.26 \times 10^4$</td>
<td>$3.56 \times 10^4$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>B</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$Time_{b\text{ound}}$ (sec.)</td>
<td>0.073</td>
<td>0.107</td>
<td>0.287</td>
<td>0.634</td>
</tr>
<tr>
<td>$Time_{\text{true}}$ (sec.)</td>
<td>0.041</td>
<td>0.874</td>
<td>3.125</td>
<td>1.910</td>
</tr>
<tr>
<td>$B^{-1}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(\lambda \sum_{l=1}^r \sigma_l)^{1/2}$</td>
<td>$1.68 \times 10^4$</td>
<td>$4.74 \times 10^4$</td>
<td>$1.33 \times 10^4$</td>
<td>$3.77 \times 10^4$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>B^{-1}</td>
<td></td>
<td>_F$</td>
</tr>
<tr>
<td>$Time_{b\text{ound}}$ (sec.)</td>
<td>0.068</td>
<td>0.158</td>
<td>0.308</td>
<td>0.840</td>
</tr>
<tr>
<td>$Time_{\text{true}}$ (sec.)</td>
<td>0.293</td>
<td>1.179</td>
<td>1.286</td>
<td>1.518</td>
</tr>
</tbody>
</table>

Table 2.2: Table illustrating the tightness of the upper bound in the lemma for the operators of the BVP.
Reduction of $\mathbb{Y} = \mathbb{B}^{\ast} \mathbb{B}$, with $M = 32$ and initial separation rank $r_\mathbb{Y} = 2025$.

<table>
<thead>
<tr>
<th>Deterministic</th>
<th>Deterministic + randomized</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to compute $</td>
<td></td>
</tr>
<tr>
<td>Value of $</td>
<td></td>
</tr>
<tr>
<td>$r_\mathbb{Y}$ after reducing by gALS</td>
<td>21</td>
</tr>
<tr>
<td>Time for reduction (sec.)</td>
<td>33.98</td>
</tr>
<tr>
<td>Rel. err. after reduction</td>
<td>$8.9 \times 10^{-4}$</td>
</tr>
<tr>
<td>(intentionally blank)</td>
<td>Time to reduce further via gALS (sec.)</td>
</tr>
</tbody>
</table>

| Final error | $8.9 \times 10^{-4}$ | 7.7 $\times 10^{-4}$ |
| Final separation rank | 21 | 24 |
| Total computational time (sec.) | 80.39 | 19.75 |

Reduction of $\mathbb{Y} = \mathbb{B}^{\ast} \mathbb{B}$, with $M = 64$ and initial separation rank $r_\mathbb{Y} = 1600$.

<table>
<thead>
<tr>
<th>Deterministic</th>
<th>Deterministic + randomized</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to compute $</td>
<td></td>
</tr>
<tr>
<td>Value of $</td>
<td></td>
</tr>
<tr>
<td>$r_\mathbb{Y}$ after reducing by gALS</td>
<td>17</td>
</tr>
<tr>
<td>Time for reduction (sec.)</td>
<td>88.60</td>
</tr>
<tr>
<td>Rel. err. after reduction</td>
<td>$9.0 \times 10^{-4}$</td>
</tr>
<tr>
<td>(intentionally blank)</td>
<td>Time to reduce further via gALS (sec.)</td>
</tr>
</tbody>
</table>

| Final error | $9.0 \times 10^{-4}$ | 8.0 $\times 10^{-4}$ |
| Final separation rank | 17 | 18 |
| Total computational time (sec.) | 196.82 | 33.37 |

Reduction of $\mathbb{Y} = \mathbb{B}^{\ast} \mathbb{B}$, with $M = 128$ and initial separation rank $r_\mathbb{Y} = 1600$.

<table>
<thead>
<tr>
<th>Deterministic</th>
<th>Deterministic + randomized</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to compute $</td>
<td></td>
</tr>
<tr>
<td>Value of $</td>
<td></td>
</tr>
<tr>
<td>$r_\mathbb{Y}$ after reducing by gALS</td>
<td>-</td>
</tr>
<tr>
<td>Time for reduction (sec.)</td>
<td>-</td>
</tr>
<tr>
<td>Rel. err. after reduction</td>
<td>-</td>
</tr>
<tr>
<td>(intentionally blank)</td>
<td>Time to reduce further via gALS (sec.)</td>
</tr>
</tbody>
</table>

| Final error | - | 7.5 $\times 10^{-4}$ |
| Final separation rank | - | 12 |
| Total computational time (sec.) | - | 73.90 |

Table 2.3: Comparison of deterministic and randomized rank reduction of $\mathbb{B}^{\ast} \mathbb{B}$. The deterministic reduction for $M = 128$ could not be performed within the limits of physical memory (\sim 4GB).
Appendix A

Keeping Greed Good: Sparse Regression Under Design Uncertainty with Application to Biomass Characterization Data
A.1 Introduction

(Note: This work was co-authored by Ryan Elmore and Wesley Jones of the National Renewable Energy Laboratory. It is currently under review.)

This paper is motivated by the practical problem of how to meaningfully perform sparse regression when the predictor variables are observed with measurement error or some source of uncertainty. We will refer to this error or noise as design uncertainty to emphasize that perturbations in the design matrix may arise from a number of random sources unrelated to experimental or measurement error per se. Recent work in this area has just begun to address the issue of sparse regression under design uncertainty from a theoretical point of view. We are primarily interested in describing an approach that, while theoretically justifiable, is essentially pragmatic and broadly applicable. In short, we argue that greed - a basic feature of many sparsity promoting algorithms - is indeed good [135], so long as the design data is scaled by the uncertainty variances. We demonstrate the efficacy of scaling from several points of view and validate it empirically with a biomass characterization data set using two of the most widely used sparse algorithms: least angle regression (LARS) and the Dantzig selector (DS).

Our work was motivated by an example from a biomass characterization experiment related to work at the National Renewable Energy Laboratory. The example is described in detail in Section A.4 and contains repeated measurements of mass spectral (design, or predictor) and sugar mass fraction (response) values within each switchgrass sample. The domain scientists' goal was to find a small subset of masses in the spectrum that could be used to predict sugar mass fraction. We will show that the replication of each measurement allows for simple estimates of the error variances which, in turn, may be used to guide the model selection procedure. Thus, we are interested in sparse regression under design uncertainty. We would also like for a scientist examining the model to have some hope of interpreting its meaning, either for immediate understanding or to indicate new research directions.

Sparse regression by $l_1$ minimization is a thriving and relatively young field of research. In the statistical inference literature, early stepwise-type algorithms paved the way for the now-familiar lasso [133], least angle regression (LARS) [48], and many variants tailored to specific problems (for example, [151, 155,
A parallel evolution in the signal processing literature led to the development of widely used basis and matching pursuit algorithms [35, 36, 135], the Dantzig selector (DS) [31], and many others (see [49], Chapters 3 and 5, for a good overview). Despite their mostly independent development, the algorithms coming out of the statistical and signal processing worlds lead to remarkably similar results in many applications (e.g., [21]).

Linear regression under the assumption of design uncertainty has, in comparison, a long history, going by various names such as error in variables or functional modeling, and a variety of techniques have been developed to address it (e.g., [58, 54, 55]). Until fairly recently, however, much of the analysis of sparse representations has not confronted this issue. As we will discuss, there is good reason for this, namely, that it obfuscates the goal of sparse regression.

Several recent works that have looked at sparse regression under various assumptions about the noise should be mentioned. In [119], the authors develop a Dantzig-like estimator that they argue is more stable than the standard lasso or Dantzig. The authors in [128] describes an algorithm to estimate the lasso solution and the noise level simultaneously. A similar idea, leading to the “adaptive lasso”, was developed in [78] under homoscedastic assumptions. An algorithm that hybridizes total least squares [59], a computational error in variables model, and the lasso was also recently published [153].

The work that comes nearest to our discussion is by Wagener and Dette [141, 142]. In these papers, the authors present some asymptotic results for bridge and lasso estimators under the assumption of heteroscedasticity. In particular, they develop a weighting scheme that leads to adaptive lasso estimates that are sign consistent (i.e., they satisfy the “oracle property”).

We consider this paper to be somewhat disjoint from the aforementioned for two reasons. First, we are primarily concerned with an approach that incorporates empirical knowledge of design uncertainty into the analysis. Second, we wish to argue from a more general, and necessarily more heuristic, point of view that does not require stringent conditions, such as those in [141], Section 3, to hold. In other words, we want to allow for the possibility that the data that is given to us may be “messy.” For example, we do not expect the design matrix to satisfy the restricted isometry property or to have low mutual coherence which, under certain circumstances, would guarantee the efficacy of an appropriate sparse algorithm.
A central notion throughout this paper is that many of the standard sparse regression algorithms are greedy, that is, they search for a solution incrementally, using the best available update at any given point in the search. As such, we argue that estimates of uncertainty should modify the notion of greed. Some algorithms, such as orthogonal matching pursuit (OMP), basis pursuit (BP), and forward stagewise regression (FS), are explicitly greedy. Others, like those that solve the lasso and Dantzig selector problems, may also be viewed as greedy via their connection to homotopy methods [8, 9, 48]. These methods generally take an initial estimate of the solution and move along a continuous path to the final one, choosing the best available search direction at each step.

Initially, we take forward stagewise (FS) regression as our prototype for analysis, noting its close relationship to the lasso and LARS [74], as well as OMP and BP [49]. We show that for all solution paths of a fixed norm, the uncertainty of the residual and the solution norm have a dual-like relationship in which the homogeneity of one induces inhomogeneity of the other, and that one can move from one problem to the other via a scaling of the design variables. From the standpoint of sparse pursuit, we argue that, as a general principle, uniform growth of the uncertainty along the solution path is preferable to uniform growth of the solution norm. Similar arguments are shown to apply to the Dantzig selector (DS). We then compare LARS and DS cross-validated model selection on a repeated measures biomass characterization data set in which variances are estimated via an analysis of variance (ANOVA) model. In this application, scaling by the uncertainty variance leads to sparser and more accurate models. Prediction error is reduced even further if, after down-selection of the variables by LARS and DS, the solution is updated via an $l_2$ method such as ridge regression.

A.2 Regression under design uncertainty

In this section, we formulate the model of interest and outline the challenges posed by design uncertainty. More importantly, we derive a simple estimate of this quantity which will play a central role in the discussion. We also give a simple example that illustrates how a very sparse solution can sometimes be associated with more of the design uncertainty than a less sparse one, further motivating our approach.
A.2.1 Model

We consider response data of the form,

\[ y_i = w_i + \epsilon_i, \]  
\[ w_i \sim N(0, \sigma_w^2), \]  
\[ \epsilon_i \sim N(0, \sigma_\epsilon^2), \]  
\[ \text{Cov}(w, \epsilon) = 0, \]

and design data,

\[ x_{ij} = v_{ij} + \delta_{ij}, \]  
\[ v_{ij} \sim N(0, \sigma_v^2), \]  
\[ \delta_{ij} \sim N(0, \sigma_\delta^2), \]  
\[ \text{Cov}(v_j, \delta_j) = 0, \quad \forall j \]

for \( 1 \leq i \leq n \) and \( 1 \leq j \leq p \). The assumptions on \( w \) and \( v_j \) imply that the data are mean centered, and we interpret \( \epsilon \) and \( \delta_j \) as independent uncertainties,

\[ \text{Cov}(\epsilon, \delta_j) = 0, \quad \forall j, \]  
\[ \text{Cov}(\delta_j, \delta_k) = 0, \quad j \neq k, \]

arising from measurement error, natural within-sample variability, or other random sources.

We will often express the system in matrix form,

\[ Y = W + \epsilon, \]  
\[ X = V + \Delta, \]

where \( \epsilon = (\epsilon_1, \epsilon_2, \ldots, \epsilon_n)' \) and \( \Delta = [\delta_{ij}]_{n \times p} \), and take the columns of \( X \) and \( Y \) to be scaled to unit variance, leading to the constraints

\[ \sigma_w^2 + \sigma_\epsilon^2 = 1, \]  
\[ \sigma_v^2 + \sigma_\delta^2 = 1, \quad \forall j. \]
Furthermore, we have

\[ \text{Cov}(X) = \text{Cov}(V) + \Sigma^2, \]

where \( \Sigma^2 \equiv \text{Cov} (\Delta) = \text{diag}(\sigma^2_{\delta_1}, \sigma^2_{\delta_2}, \ldots, \sigma^2_{\delta_p}) \) by the independence of the errors. When using finite sample estimates \( s^2_{\delta_j} \) of \( \sigma^2_{\delta_j} \), we denote the corresponding matrix by \( S^2 \). In the absence of noise \( (\sigma^2_{\epsilon} = 0 \text{ and } \sigma^2_{\delta_j} = 0, \forall j) \), we assume that the design and response admit a linear model,

\[ w = V\beta. \tag{A.3} \]

We are particularly interested in the case where \( \beta \) is sparse: loosely speaking, many of its elements are zero.

In the application discussed in Section A.4, repeated measurements are used to estimate the variances \( \sigma^2_{\delta_j} \). For a more theoretical application, it may be the case that these parameters are known exactly. Either way, for the remainder of the discussion we assume that either the variances or their sample estimates are available.

### A.2.2 The challenge of design uncertainty

One intrinsic challenge in working with noisy design data is that the estimated regression coefficients are attenuated from their true values. Suppose, instead of (A.3), we were to solve

\[ y = X\beta \tag{A.4} \]

via ordinary least squares (OLS) to obtain \( \hat{\beta} \). For \( p = 1 \), it is straightforward to show that

\[ \frac{\mathbb{E}\{\hat{\beta}\}}{\beta} = \frac{\sigma^2_{\nu}}{\sigma^2_{\epsilon} + \sigma^2_{\delta}} < 1, \tag{A.5} \]

where \( \mathbb{E} \) denotes expected value. This implies that the estimators are biased towards zero by an amount that depends on the signal-to-noise ratio, \( \sigma^2_{\epsilon}/\sigma^2_{\delta} \). More generally, for any full rank \( X \in \mathbb{R}^{n \times p} \) with \( n \geq p \), \( X \) may be diagonalized such that in the new system of coordinates an analogous result holds.

Design uncertainty also degrades the model fit even if the exact solution \( \beta \) is known. To see this,
consider the residual error when design uncertainty is present:

\[
E\{||y - X\beta||_2^2\} = E\{||(w + \epsilon) - (V + \Delta)\beta||_2^2\} \\
= E\{||\epsilon - \Delta\beta||_2^2\} \\
= E\{\epsilon'\epsilon - 2\epsilon'\Delta\beta + \beta'\Delta'\Delta\beta\} \\
= \sigma_\epsilon^2 + \beta'\Sigma\{\Delta'\Delta\}\beta \\
= \sigma_\epsilon^2 + \beta'\Sigma^2\beta \\
= \sigma_\epsilon^2 + ||\Sigma\beta||_2^2,
\]

where we have explicitly used (A.3) and the independence of the errors.

This brings us to a main point, that the contribution of the design uncertainty to the residual is of the form \(||\Sigma\beta||_2^2\), which is quadratic in \(\Sigma\). While we may only have access to the attenuated estimate \(\hat{\beta}\) of \(\beta\), the structure of the residual error remains the same with respect to the error variances. We illustrate the effect this can have on sparse regression with a simple example.

Suppose \(p = 3\) and

\[
\begin{align*}
\mathbf{w} &= v_1, \\
v_1 &= \frac{1}{\sqrt{2}}(v_2 + v_3), \\
\sigma_\epsilon^2 &= 0, \\
\Sigma^2 &= \text{diag}(\frac{1}{2}, \frac{1}{4}, \frac{1}{4}).
\end{align*}
\]

The system admits the two solutions \(\beta^{(1)} = (1, 0, 0)'\) and \(\beta^{(2)} = \frac{1}{\sqrt{2}}(0, 1, 1)\). The first solution is the sparsest but in light of (A.6) has greater expected error since \(||\Sigma\beta^{(1)}||_2 = 1/2\) while \(||\Sigma\beta^{(2)}||_2 = \sqrt{2}/4\). Hence, recovery of the sparsest solution results in greater uncertainty in the fit than the less sparse one. The issue becomes even more prominent - and more difficult to track - in higher dimensions with non-trivial covariance of the design matrix.

Apparently, *greed is not always good under design uncertainty.*
A.3 Scaling penalizes design uncertainty in the solution path

In this section, we briefly describe a prototypical greedy algorithm for sparse regression, forward stagewise regression (FS). We do so because it is helpful to have a particular algorithm in mind for the discussion, and this one is particularly easy to understand. In addition, it solves the widely-used lasso optimization problem and thus is closely related to a variety of other important algorithms [48, 74]. Next, we state the main result and provide simple algebraic and geometric interpretations of it. Finally, we note implications of the result for the Dantzig selector problem.

A.3.1 A prototypical pursuit algorithm: forward stagewise (FS) regression

The FS algorithm may be summarized as follows:

1. Fix small $\gamma > 0$ and initialize: $\hat{\beta} = 0$, $r = y$.

2. Identify the design variable $x_j$ most correlated with $r$.

3. Incremental update: $\hat{\beta}_j \leftarrow \hat{\beta}_j + \eta_j$, where $\eta_j = \gamma \cdot \text{sign}(\text{Corr}(x_j, r))$.

4. Subtract the projection of $r$ onto $x_j$: $r \leftarrow r - \eta_j x_j$.

5. If the residual norm is small enough, stop. Otherwise, return to step 2.

Qualitatively, the algorithm finds the best search direction - the coordinate with highest residual correlation - and takes a small step in that direction. It does so iteratively, updating the solution and residual at each step, until the minimal residual error is reached.

As an optimization procedure, FS regression (like the lasso and LARS) implicitly solves

$$\arg \min_{\beta \in \mathbb{R}^p} \frac{1}{2} ||y - X\beta||_2^2 \text{ subject to } ||\beta||_1 < \lambda, \quad (A.7)$$

which is often expressed in Lagrangian form,

$$\arg \min_{\beta \in \mathbb{R}^p} \frac{1}{2} ||y - X\beta||_2^2 + \lambda ||\beta||_1, \quad (A.8)$$

for a range of values of the tuning parameter $\lambda > 0$. In the limit $\lambda \to 0$, the optimum is attained by the ordinary least squares solution, while solutions for $\lambda \to \infty$ are increasingly sparse ($\lambda = \infty \iff \beta = 0$) [133].

---

1 In LARS, the step is computed in a particularly efficient way but the final solution path is essentially the same.
A.3.2 Main result

Our main result is simple: it says that for all solutions of a fixed norm, the accumulated design uncertainty (estimated by $||\hat{\Sigma} \hat{\beta}||_2$ in equation (A.6)) is path-dependent unless the data are scaled by the uncertainty variance. In other words, scaling the data leads to a uniform increase of the design uncertainty contribution, independent of the search direction.

To see this (and with a slight abuse of notation), we first modify equation (A.4) to include scaling of the design variables,

$$y = XD^{-1} \delta,$$

noting that if $\delta$ solves (A.3), then $D\delta$ solves (A.9). The expected residual variance is then

$$E\{||y - XD^{-1} \delta||_2^2\} = \sigma^2 + E\{\delta' \Sigma^{-1} \Delta \Delta^{-1} \delta\}$$

$$= \sigma^2 + \beta' \Sigma^{-1} \delta,$$

$$= \sigma^2 + ||\Sigma D^{-1} \delta||_2^2.$$  

(A.10)

Let $D$ be a diagonal scaling matrix and let $U(\beta; D) = ||\Sigma D^{-1} \delta||_2^2$ be the design uncertainty associated with a solution $\delta$ of fixed norm, $||\delta||_2^2 = T^2$. If $D = \Sigma$ then the total uncertainty is independent of the uncertainty variances,

$$\frac{\partial U}{\partial \sigma^2_{\delta_j}} = 0, \quad \forall j,$$

for any $\sigma^2_{\delta_j}$ and $T$.

The proof follows directly from (A.10). First, note that

$$U(\beta; D) = \begin{cases} ||\beta||_2^2 = T^2, & \text{if } D = \Sigma, \\ \sum_{j=1}^p \sigma^2_{\delta_j} D^{-2}_{jj} \beta^2_j, & \text{otherwise}. \end{cases}$$

Taking derivatives with respect to $\sigma^2_{\delta_j}$,

$$\frac{\partial U}{\partial \sigma^2_{\delta_j}} = \begin{cases} 0, & \text{if } D = \Sigma, \\ ||D^{-1} \delta||_2^2, & \text{otherwise}. \end{cases}$$
Scaling the data will result in solutions of different norms, so that two solutions of norm $T$ under different scalings $D_1^{-1}$ and $D_2^{-1}$ are not directly comparable in terms of the underlying optimization problem. However, the result says that scaling by $D = \Sigma$ leads to a solution space in which all solutions of identical norm have identical uncertainty.

### A.3.2.1 Algebraic interpretation of scaling

Based on our claim, we consider scaling the columns of $X$ by the associated uncertainties,

\[
D = \Sigma, \\
X \leftarrow XD^{-1}.
\]

The most obvious effect of scaling is that the correlations change and so (potentially) does the order in which the variables are selected (step 2 of the FS algorithm). Recalling that the columns of $X$ and $y$ have unit variance, we initially have

\[
\text{Corr}(x_j, y) = x_j' y,
\]

while after scaling,

\[
\text{Corr}(x_j, y) = x_j'y/\sigma_j.
\]

A less obvious effect of scaling is that the underlying problem (A.8) is transformed so that uncertainty in the solution path is penalized explicitly. The scaled problem,

\[
\arg \min_{\beta \in \mathbb{R}^p} \frac{1}{2}||y - XD^{-1}\beta||_2^2 + \lambda||\beta||_1,
\]

by a simple change of variables, $\beta \leftarrow D\beta$, may be written

\[
\arg \min_{\beta \in \mathbb{R}^p} \frac{1}{2}||y - X\beta||_2^2 + \lambda||D\beta||_1.
\]

We note that this is the “generalized lasso” problem described in [134]. The lasso penalty term represents the “$l_1$ version” of the design uncertainty (recall that $||\Sigma\beta||_2 \leq ||\Sigma\beta||_1 \leq \sqrt{p}||\Sigma\beta||_2$, by norm equivalence). Hence, scaling by $D = \Sigma$ leads to a direct $l_1$ penalization of design uncertainty within the lasso framework.
A.3.2.2 Geometric interpretation of scaling

Figure A.1: The dual-like relationship between uncertainty and solution norm. Left: Without scaling, uncertainty grows anisotropically (indicated by an ellipse around the origin) for fixed step size $\gamma$ of the greedy algorithm, while the solution norm grows uniformly (indicated by perpendicular contour lines on each axis). Right: After scaling by $D^{-1}$, the uniformity of uncertainty and solution norm are reversed. The notation $||d(\cdot)||_1$ represents the increase in the $l_1$ penalty term for a small step in the solution path.

Geometrically, scaling by the uncertainty induces a dual-like problem in which the homogeneity of solution norm and the uncertainty are reversed (Figure A.1). In particular, before scaling, a step of fixed size $\gamma$ leads to constant growth of the $l_1$ penalty but potentially non-uniform growth of the uncertainty. After scaling, on the other hand, the uncertainty grows uniformly at each step while the $l_1$ penalty does not.

Of course, for a given data set, the greedy algorithms we have discussed are not random but deterministic. But if we consider the task of sparse regression as applying to an ensemble of noisy data sets, one can think of the solution paths as being effectively random (for a similar line of reasoning see, e.g., [45]). That is, a statistical analysis of the algorithm is then necessarily and justifiably carried out in terms of expectations, rather than specific search paths.
A.3.3 Connection to the Dantzig selector

While a detailed analysis is beyond our scope, we take a brief moment to point out the connection between scaling and the Dantzig selector. Candes and Tao [31] proposed an alternative formulation for sparse regression,

$$\arg\min_{\beta} ||\beta||_1 \quad \text{subject to} \quad ||X'(y - X\beta)||_\infty \leq \lambda \sigma,$$

where $\lambda > 0$ is a tuning parameter (different from the lasso parameter) and $\sigma^2$ is the variance in (A.1). The Dantzig selector has two main features that distinguish it from other pursuit algorithms. The first is that the problem may be written explicitly as a linear program (LP), for instance,

$$\arg\min_{\alpha, \beta} 1'\alpha \quad \text{subject to} \quad -\alpha \leq \beta \leq \alpha \quad \text{and} \quad -\sigma \lambda 1 \leq X'(y - X\beta) \leq \sigma \lambda 1.$$

The second is that the $l_\infty$ constraint is with respect to residual correlations as opposed to residual error. This seems intuitively correct since, in the presence of noise, we would expect the residual corresponding to an optimal solution to have exactly this property.

Now consider the change of variables $X \leftarrow XD^{-1}$, $\beta \leftarrow D\beta$, and $\alpha \leftarrow D\alpha$ as in Section A.3.2.1. In the Dantzig context, this leads to the linear program:

$$\arg\min_{\alpha, \beta} 1'(D\alpha) \quad \text{subject to} \quad -\alpha \leq \beta \leq \alpha \quad \text{and} \quad -\sigma \lambda D1 \leq X'(y - X\beta) \leq \sigma \lambda D1.$$

Notice that the feasible region is stretched along the noisier dimensions (proportionally to $\sigma_{\delta_j}$), resulting in relaxed requirements for the residual correlation in those coordinates. This is reasonable, as we would expect the accuracy for a given variable to be inversely related to its uncertainty. As in the lasso context, scaling also results in an explicit $l_1$ penalization of the variables commensurate with their noise level via minimization of the quantity $1'(D\alpha)$.

Continuing the example from Section A.2.2, recall that

$$\Sigma^2_{11} = \frac{1}{2},$$
$$\Sigma^2_{22} = S^2_{33} = \frac{1}{4},$$
and that the uncertainty in the sparsest solution was greater than the next sparsest. Figure A.2 gives a concrete illustration of the solution path for the scaled and unscaled data as well as the uncertainty in the fit (left panel). After three FS steps (identically for lasso and LARS), there is zero residual error for both the scaled and unscaled design (red lines, right panel). However, the uncertainty associated with $\beta_2$ is less than that of $\beta_1$ by a factor of 2 (black lines, right panel).

Figure A.2: A toy example in which the sparsest solution has more uncertainty than the next sparsest one (see Section A.2.2). Left: the regression coefficients at each stage of the FS algorithm. Only non-zero coefficients are plotted. Right: the uncertainty as estimated by $||S\beta||_2$ (black), with residual sum of squares (RSS) on the right axis (red). The results are identical for lasso and LARS.

### A.4 Application to biomass characterization data

In this section, we present results for both LARS and DS applied to a biomass characterization data set, with and without scaling. We highlight the challenges in working with this data, and illustrate the efficacy of scaling.
A.4.1 Description of the data

Figure A.3: Graphical summary of the biomass characterization data. The left plot shows a typical raw MBMS spectrum. The inset shows, after pre-processing, the maximum cross-correlations for each peak and indicates a high degree of mutual dependence between the predictors. The right plot shows the distribution of variances estimated via a random one-way ANOVA model before normalization. The marker radius is proportional to mass-to-charge ratio of the peak, while the response ratio is indicated by a red triangle. A 1 : 1 ratio is indicated by a black line. Points lying further down and to the right of the solid black line have higher fidelity (i.e., higher signal-to-noise ratio).

The characterization experiment we consider is motivated by a desire to quickly and inexpensively screen potential biofuel candidates for recalcitrance - a plant’s natural resistance to releasing usable sugars - after a chemical or enzymatic pre-treatment. Here, $n = 759$ switchgrass plants were grown at different outdoor locations and in uncontrolled conditions. The predictors consist of $p = 421$ pyrolysis molecular beam mass spectral (pyMBMS or MBMS, [130]) lines measured twice for each sample. As each sample is pyrolyzed, the spectrometer counts the number of molecules that reach a detector over a range of mass-to-charge ratios. The raw spectrum for each sample is then normalized to have unit mass and each peak is divided by a standard (control) value measured during the same run, allowing samples from different experiments to be
compared directly. So, after pre-processing, each peak may be thought of as being an expression level for that mass-to-charge ratio relative to a control.

The response is the mass fraction of extractable glucose as inferred by the absorbance of 510-nm visible light, where each sample is measured in triplicate [123]. In this experiment, a previously validated linear model is calibrated via measurement of a pure glucose sample. The mass and absorbance of each biological sample from the same run are then input to the calibrated model, yielding an estimate of glucose mass fraction for that sample.

The question we ask is: can the MBMS spectrum (a proxy for chemical composition) be used to predict the mass fraction of extractable glucose (usable biofuel)? To answer this, we seek a sparse linear model that incorporates estimates of uncertainty. Brief justifications for this approach are:

- Sparse: The spectroscopy experiment results in high cross-correlations between the peaks because large masses break into smaller ones in a somewhat predictable way. Hence, we expect a significant amount of redundancy in the peaks. In addition, the relationship between mass spectral peaks and cell chemistry is complex, making a sparse model appealing in that it narrows the focus of future investigations to a few, rather than hundreds, of peaks and their associated compounds.

- Incorporates uncertainty: Some of the peaks are far noisier than others, leading to unequal uncertainties. We would like to ensure that the model depends on the noisy peaks as little as possible, without completely excluding them from consideration.

- Linear: The assumed physical model is one of linear mixture, i.e., doubling the concentration of an analyte in the sample should result in a doubling of its spectral signature.

The data are summarized graphically in Figure A.3. A typical raw mass spectrum is shown in the left panel where line height indicates count following convention for this field. The inset plot shows the maximum absolute cross-correlation of each peak with every other peak, from which we infer that there is a high degree of linear dependence among the variables, especially the smaller masses. In the right panel, the estimated total and within-sample ANOVA variances are shown before normalization or scaling, with equality indicated by a black line. The mass-to-charge ratios of the MBMS lines are proportional to the marker radius while
glucose is indicated by a triangle. Clearly, many of the peaks are quite noisy, with almost all of the variance attributed to noise.

A.4.2 Methods

Model selection was performed using nested $k$-fold cross validation (CV), in which standard $k$-fold CV errors were averaged over $100/k$ outer loops for $k = 2, 5,$ and $10$. This approach ensured that 100 different prediction models were validated for each choice of $k$. We fit both LARS and Dantzig models for comparison. LARS models were fit in R using the lars package of Efron and Hastie [73, 131], and Dantzig in MATLAB using the L1 Homotopy Toolbox of Asif and Romberg [2, 7, 9]. As has been suggested before (e.g., [49] Chapter 8.5), it can sometimes be beneficial to regress $y$ on the sparse predictor set using another fitting procedure. For comparison, we used the LARS- and Dantzig-selected peaks as input to cross validated ridge regression via the parcor package in R [89]. In all instances, the scaling matrix was estimated as part of the cross validation procedure (see Appendix A for details).

A.4.3 Results

Cross-validation results are given in Table A.2 in Appendix B, and may be summarized as follows. Scaling leads to:

(1) improved accuracy, as measured by cross-validated MSEP, for both LARS and DS (Figure A.4)

(2) increased sparsity for both LARS and DS (Figure A.5)

(3) higher degree of consistency between LARS and DS
Figure A.4: LARS cross validation results for $k = 10$. The left plot shows the mean square error of prediction (MSEP) in bold lines and standard error as shaded regions. The dashed lines are estimates of uncertainty via $||\hat{S}\hat{\beta}||_2$, with units on the right axis. The optimal model is indicated with an x. Apparently, scaling by uncertainty variances leads to a sparser and more accurate model, with less associated uncertainty. The right plot is identical to the one in Figure A.3, but with solution coordinates selected by LARS given different markers based on scaling (blue circles: unscaled, red diamonds: scaled). At least 4 peaks with high signal-to-noise are clearly selected after scaling that are not otherwise (green arrows, with numbers indicating the $m/z$ ratio).

The left panel of Figure A.4 shows the prediction error per LARS step (solid lines), the standard error (shaded regions), and the uncertainty as estimated by $||\hat{S}\hat{\beta}||_2$ for $k = 10$ (dashed lines). The optimal models are indicated by x’s. While the standard error of prediction is similar for the scaled and unscaled case, the uncertainty accumulates more slowly for the scaled input (almost identical results hold for $k = 2, 5$, not shown). The right panel provides a graphical impression of the quality of the variables selected for the scaled and unscaled data. One can see that, in general, the scaled approach (red diamonds) leads to selection of peaks with higher signal-to-noise ratio, indicated by green arrows, than the unscaled (blue circles).
Figure A.5: Number of non-zero coefficients for LARS and DS for \( k = 2, 5, 10 \). In all cases, the prediction error decreases as \( k \) (i.e., the number of training samples) increases. Without scaling, the more accurate models use more variables while with scaling, remarkably, they use fewer.

Figure A.5 shows the sparsity of the LARS and DS models as a function of the CV fold sizes. Remarkably, the number of non-zero coefficients for both LARS and DS actually increases with increasing sample size when the data are unscaled. This is somewhat surprising since, heuristically, one would expect the model selection to be more discriminating as more samples are utilized. On the other hand, the number of non-zeros decreases with increasing sample size for the scaled data. To explain this, we speculate that when the data are unscaled, it is more likely for the algorithm to select variables that are either neutral or even detrimental with respect to prediction. If this is the case, then our results suggest that scaling leads to a more discriminating variable selection and higher prediction accuracy.

Finally, while the LARS and DS solutions are not in perfect agreement in either case (scaled or unscaled), they are seen to be in better agreement after scaling. Only 46% of the LARS peaks are also selected by DS without scaling, while the number is 86% with scaling. Alternatively, of the total number of distinct peaks selected by LARS and DS combined, only 24% are common to both without scaling, while 45% are common to both with scaling.
A.4.4 Discussion

It should be stated up front that the assumption of linearity made in Section A.4.1 does not appear to be completely valid. While the assumption should be valid on physical grounds, there are obviously experimental, biological or other factors that introduce significant error terms beyond those formulated in Section A.2.1. That said, the fact that scaling leads to a reduction in CV error, increased sparsity, and better agreement between LARS and DS suggests that the method can still be practically useful under non-ideal circumstances.

While some of the peaks identified by both scaled LARS and DS have been previously recognized as related to recalcitrance, many have not (see Appendix C). Of particular interest are the peaks with large m/z ratios, as these are less likely to be correlated coincidentally with recalcitrance: light particles can originate from a variety of sources, but less so for larger particles. Furthermore, some of the unknown peaks have regression coefficients that are not small. We believe that these results warrant taking a further look at the unknown peak associations to better understand chemical mechanisms of recalcitrance.

A.5 Conclusions

We have argued that sparse regression under design uncertainty presents several challenges that (to the best of our knowledge) have not been addressed in the literature. Focusing on the uncertainty term, $\|\Sigma \hat{\beta}\|_2^2$, in the residual error (A.6), we propose a scaling of the design variables by their uncertainty variances. In the context of greedy algorithms, doing this guarantees a uniform growth of uncertainty regardless of the order in which the variables are selected. Within the lasso formulation, scaling is shown to enforce an $l_1$ penalization of the uncertainty. In the Dantzig selector context, scaling leads to modified bounds on the residual error that reflect the amount of uncertainty associated with each variable.

In a biomass characterization application, scaling is shown empirically to reduce uncertainty in the optimal solution. It also leads to sparser solutions and lower prediction error. The solution estimates are improved even further if the LARS- and Dantzig-selected peaks are used independently for ridge regression. In addition, these models are more consistent with one another after scaling, that is, they identify more of the
same predictors. The improvements resulting from scaling are promising and deserve further consideration.

Acknowledgements

The authors would like to thank Terry Haut for many useful conversations, Peter Graf for his critical eye, and Matthew Reynolds for help with proofreading. This work was supported by the DOE Office of Biological and Environmental Research through the BioEnergy Science Center (BESC). BioEnergy Science Center (BESC) is a US Department of Energy Bioenergy Research Center supported by the Office of Biological and Environmental Research in the DOE Office of Science.
Appendix A

**ANOVA model.** We use a one way, random effects ANOVA model to estimate the uncertainty variances. Let \( r \) denote the number of replicate measurements of a random variable \( Z \) and let \( n \) denote the number of samples. The relevant quantities needed to estimate the variance components are shown in Table A.1. In particular, the estimates are \( s^2_\delta = \frac{\text{SSE}}{\text{df}} \), and \( s^2_\nu = \frac{(\text{SSTr}/\text{df}) - \sigma^2_\delta}{r} \).

<table>
<thead>
<tr>
<th>Source</th>
<th>df</th>
<th>Sum of squares</th>
<th>Expected mean square</th>
</tr>
</thead>
<tbody>
<tr>
<td>Treatment</td>
<td>( n - 1 )</td>
<td>( \text{SSTr} = \sum_{i=1}^{n} r(\bar{z}<em>i - \bar{z}</em>\cdot)^2 )</td>
<td>( r\sigma^2_\nu + \sigma^2_\delta )</td>
</tr>
<tr>
<td>Error (uncertainty)</td>
<td>( n(r - 1) )</td>
<td>( \text{SSE} = \sum_{i=1}^{n} \sum_{j=1}^{r} (z_{ij} - \bar{z}_i)^2 )</td>
<td>( \sigma^2_\delta )</td>
</tr>
</tbody>
</table>

Table A.1: Standard one-way, random effects ANOVA table.

**Cross validation procedure.** For clarity, we outline our procedure for cross-validated model selection using replicated measurements. It is a completely standard cross-validation procedure with the simple addition that we estimate the uncertainty variances only from the training data.

For each of the \( k \) cross-validation groups:

1. Split the data into training, \( \{y_{\text{train}}, X_{\text{train}}\} \), and test sets, \( \{y_{\text{test}}, X_{\text{test}}\} \), of appropriate sizes.

2. Using only \( X_{\text{train}} \), estimate the error variances, \( s^2_{\delta_j} \), via a suitable method (we used one-way, random effects ANOVA).

3. Form the diagonal matrix \( D_{jj} = \sigma^2_{\delta_j} \) and scale the training data, \( X_{\text{train}} \leftarrow X_{\text{train}}D^{-1} \).

4. Fit the desired models to \( y_{\text{train}} \) using scaled \( X_{\text{train}} \).

5. Using \( D \) from step 3, scale the test data, \( X_{\text{test}} \leftarrow X_{\text{test}}D^{-1} \), and predict.
Appendix B: Numerical results

<table>
<thead>
<tr>
<th>Model selection method</th>
<th>Scaling</th>
<th>$k$</th>
<th># predictors</th>
<th>MSEP</th>
<th>Avg # predictors</th>
<th>Avg MSEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>LARS</td>
<td>NO</td>
<td>2</td>
<td>47</td>
<td>0.564</td>
<td>54.3</td>
<td>0.555</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>57</td>
<td>0.551</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>59</td>
<td>0.550</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LARS</td>
<td>YES</td>
<td>2</td>
<td>36</td>
<td>0.539</td>
<td>31.7</td>
<td>0.533</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>31</td>
<td>0.531</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>28</td>
<td>0.530</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LARS-RR</td>
<td>NO</td>
<td>2</td>
<td>47</td>
<td>0.508</td>
<td>54.3</td>
<td>0.493</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>57</td>
<td>0.485</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>59</td>
<td>0.485</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LARS-RR</td>
<td>YES</td>
<td>2</td>
<td>36</td>
<td>0.492</td>
<td>31.7</td>
<td>0.492</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>31</td>
<td>0.491</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>28</td>
<td>0.494</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dantzig selector</td>
<td>NO</td>
<td>2</td>
<td>40</td>
<td>0.622</td>
<td>66.7</td>
<td>0.599</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>80</td>
<td>0.583</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>80</td>
<td>0.574</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dantzig selector</td>
<td>YES</td>
<td>2</td>
<td>76</td>
<td>0.547</td>
<td>60.3</td>
<td>0.536</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>56</td>
<td>0.528</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>49</td>
<td>0.523</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dantzig-RR</td>
<td>NO</td>
<td>2</td>
<td>40</td>
<td>0.512</td>
<td>66.7</td>
<td>0.494</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>80</td>
<td>0.487</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>80</td>
<td>0.482</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dantzig-RR</td>
<td>YES</td>
<td>2</td>
<td>76</td>
<td>0.460</td>
<td>60.3</td>
<td>0.455</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>56</td>
<td>0.454</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>49</td>
<td>0.451</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ridge regression</td>
<td>NO</td>
<td>2</td>
<td>421</td>
<td>0.533</td>
<td>421</td>
<td>0.535</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>421</td>
<td>0.536</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>421</td>
<td>0.535</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ridge regression</td>
<td>YES</td>
<td>2</td>
<td>421</td>
<td>0.515</td>
<td>421</td>
<td>0.516</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>421</td>
<td>0.517</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>421</td>
<td>0.517</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table A.2: Results of $k$-fold cross validation (see also Figure A.5). The -RR suffix indicates ridge regression was performed on the subset selected by the corresponding sparse algorithm. For comparison, results for ridge regression using all of the predictor variables are also shown.
Table A.3: Peaks identified by both scaled LARS and DS for $k = 10$. The peaks previously identified in [129] as significant to sugar release are described where possible. The LARS and DS regression coefficients were averaged and divided by the maximum to highlight the relative significance and sign of correlation of the peaks. Some of the most highly-weighted variables have not previously been identified as being related to recalcitrance.

<table>
<thead>
<tr>
<th>$m/z$</th>
<th>Assignment in [129]</th>
<th>Avg. coefficient (relative to max)</th>
</tr>
</thead>
<tbody>
<tr>
<td>45</td>
<td>?</td>
<td>+1.0000</td>
</tr>
<tr>
<td>60</td>
<td>C5 sugars</td>
<td>-0.0534</td>
</tr>
<tr>
<td>120</td>
<td>Vinylphenol</td>
<td>-0.2519</td>
</tr>
<tr>
<td>126</td>
<td>?</td>
<td>-0.1432</td>
</tr>
<tr>
<td>128</td>
<td>?</td>
<td>-0.0482</td>
</tr>
<tr>
<td>129</td>
<td>?</td>
<td>-0.2177</td>
</tr>
<tr>
<td>137</td>
<td>Ethylguaiacol, homovanillin, coniferyl alcohol</td>
<td>-0.7648</td>
</tr>
<tr>
<td>143</td>
<td>?</td>
<td>+0.1795</td>
</tr>
<tr>
<td>144</td>
<td>?</td>
<td>-0.2338</td>
</tr>
<tr>
<td>150</td>
<td>Vinylguaiacol</td>
<td>+0.8777</td>
</tr>
<tr>
<td>159</td>
<td>?</td>
<td>+0.1016</td>
</tr>
<tr>
<td>160</td>
<td>?</td>
<td>+0.1988</td>
</tr>
<tr>
<td>164</td>
<td>Allyl ± propenyl guaiacol</td>
<td>-0.3966</td>
</tr>
<tr>
<td>168</td>
<td>4-Methyl-2, 6-dimethoxyphenol</td>
<td>-0.3949</td>
</tr>
<tr>
<td>175</td>
<td>?</td>
<td>+0.0554</td>
</tr>
<tr>
<td>182</td>
<td>Syringaldehyde</td>
<td>-0.0734</td>
</tr>
<tr>
<td>194</td>
<td>4-Propenylsyringol</td>
<td>-0.8661</td>
</tr>
<tr>
<td>208</td>
<td>Sinapyl aldehyde</td>
<td>-0.0038</td>
</tr>
<tr>
<td>210</td>
<td>Sinapyl alcohol</td>
<td>+0.4334</td>
</tr>
<tr>
<td>226</td>
<td>?</td>
<td>+0.3021</td>
</tr>
<tr>
<td>264</td>
<td>?</td>
<td>+0.1883</td>
</tr>
<tr>
<td>287</td>
<td>?</td>
<td>-0.1646</td>
</tr>
<tr>
<td>371</td>
<td>?</td>
<td>-0.1970</td>
</tr>
<tr>
<td>374</td>
<td>?</td>
<td>+0.2153</td>
</tr>
</tbody>
</table>
Appendix B

Analysis of Governing Factors for Photovoltaic Loss Mechanism of n-CdS/p-CdTe Heterojunction via Multi-way Data Decomposition
B.1 Introduction

(Note: This work was co-authored by Changwon Suh, David Albin, and Rebekah Graham of the National Renewable Energy Laboratory. It is currently under review).

The crux in developing advanced solar devices is ensuring higher cell efficiency (\(\eta\)) with cost-effective manufacturing processes. In the case of CdS/CdTe heterojunction devices, for instance, the cell conversion efficiency of 17.3% has been achieved through current high-end process controls [53], while its theoretical and practical limits are 29% and 20%, respectively [95, 124]. Apart from further increasing the efficiency, however, equally crucial to device improvement is the stability of the devices, because optimal device performance does not guarantee optimal durability [5, 83].

There are at least three challenges to understanding cell degradation. The first is that the underlying physical mechanisms are inherently complex and must be balanced with design specifications. A concrete example that highlights this challenge is the inclusion of Cu in the back contact materials of CdS/CdTe solar cells. While the presence of Cu ensures a non-rectifying contact, it also causes deterioration of the long-term stability of such cells due to the fast diffusion of Cu\(^{+}\), a slow-diffusing component Cu\(_{Cd}\), and/or associated complexes in the CdTe layer, leaving behind a Cu-depleted rectifying back contact and forming Cu-related recombination centers [140, 43]. Thus, a design modification may lead to degradation in a complex and unforeseen way.

The second challenge arises in the analysis of the characterization data. For instance, current-voltage (J-V) measurements made for the study of cell stability are multivariate: they include open-circuit voltage (\(V_{oc}\)), short-circuit current density (\(J_{sc}\)), fill factor (FF), series resistance (\(R_s\)), and shunt resistance (\(R_{sh}\)) with respect to different stress factors like time, temperature, or humidity. If the J-V parameters are characterized over all combinations of stress factors, the data set can become very large [61]. Capacitance-voltage (C-V) measurements are similarly complex and are often even less well understood than J-V in terms of how they characterize device performance [145, 109].

The third challenge is the integration of data sets that are both large and heterogeneous. While difficult, this step may be key to a successful, comprehensive study of devices because it represents a synthesis
of knowledge from disparate sources [127]. A prototypical example of this is seen in the analysis of the Schottky barrier diode [154]. Information about barrier formation is available from J-V characterization, while carrier transport information is accessible via C-V characterization. However, responses of the electric field in terms of the Schottky barrier height are identifiable only by considering J-V and C-V characteristics in tandem.

In this paper, we propose an automated approach to better understanding photovoltaic (PV) loss mechanisms from reliability data generated from J-V and C-V measurements. We do so by assuming the data comes with a certain structure and applying a statistical algorithm that accommodates it. The method allows us to analyze all of the available data simultaneously while reducing the complexity of how it is represented, and is computationally quick (for the data set we consider, the algorithm finishes in seconds on a standard laptop computer). This approach leads to a model that is comprehensive and easily interpretable; it is also more systematic than manual inspection of the data or methods that examine subsets of the data independently.

B.2 Experimental

In this section, we describe the experimental data to be analyzed and illustrate the difficulty in visualizing the data and identifying global trends “by eye”, thus motivating our approach. A complete and detailed descriptions of experiments and their applications can be found in [61].

B.2.1 Fabrication and measurement of CdS/CdTe cells

For accelerated lifetime testing, five laboratory CdS/CdTe cell devices, each having about 4 µm CdTe layer thickness, were stressed under conditions of one-sun illumination, open-circuit bias, and an acceleration temperature of approximately 60~65 °C for durations exceeding 1000 hours. The C-V method was performed using an Agilent 4294A Precision Impedance Analyzer operated at 100 kHz with a 50 mV oscillation voltage.

The order of the scan direction can sometimes lead to different measurement outcomes [6]. As such, the C-V parameters were measured in both the forward and reverse scan directions, and their difference, which we refer to as the “hysteresis” scan value, was also calculated. By forward and reverse voltage scans
we mean the direction that the voltage was being swept during the measurement. All reverse scans were made from +0.49 V to -1.49 V by increments of -0.01 V, followed by a 5 minute voltage bias at -1.5 V, and then followed by a forward scan back to +0.49V. Here, the voltage bias was considered as one of the crucial parameters in our analysis because it is a proxy related with the depletion width \( W_d \) providing spatial information of the cells [61].

Hysteresis is often non-zero due to the presence of deep states as well as mobile ions [6, 50]. In the ‘bi-direction’ scan, potential damage from forward current heating is minimized by performing it at room temperature and maintaining cell bias constant only during negative voltages. In addition, a quick scan speed was used to minimize relaxation effects during the voltage scan. Figure B.1 shows PV loss behavior of the cell devices using \( FF \) calculated from J-V measurement as well as exemplary C-V results in terms of stress time.

**B.2.2 Difficulty in visualizing global trends**

Figure B.2 shows several visualization cross sections of the C-V data which is intrinsically four-way (we describe the four-way format in Section B.4.1). It highlights the difficulty of trying to surmise data-wide trends by visualization alone. In the top two panels (Figure B.2 (a) and (b)), hysteresis of net carrier density \( N_{a,\text{hyst}} \) and depletion width \( W_{d,\text{hyst}} \) are averaged over all devices. For each measured stress time and voltage bias, contours of the average value are indicated by color. This is done so that gross features, for a fixed scan direction, may be captured by eye. One can see that, for example, \( N_{a,\text{hyst}} \) is markedly lower at positive voltages than at negative ones, although less variable. In contrast, \( W_{d,\text{hyst}} \) appears to be less variable, with an average maximum around +0.1V. For comparison, the lower two panels of Figure B.2 ((c) and (d)) show another cross section of the data. Here, \( N_a \) is shown for hysteresis and forward scan directions at all voltage biases. Red regions draw the eye towards the combination of carrier density and voltage bias corresponding to high stress (long stress times).

Although some trends are discernible, it is clear that two-dimensional visualization is probably insufficient for capturing general, multivariate trends in a convincing way. This is one of the primary motivations for a more quantitative approach that can handle complex data structures.
Figure B.1: Exemplary data representation generated from (a) J-V measurement, (b-d) C-V measurement of CdS/CdTe cell devices in this study. The bars in (c) are on a projected plane of stress time and \( W_{d_{\text{hyst}}} \). The noise in the negative bias regime of (d) stems from the standard numerical scheme used to estimate net acceptor density (\( N_a \)) from capacitance (C).

Figure B.2: Visualization cross sections of the data. The difficulty in interpretation dramatically increases when simultaneously considering all stress times, C-V parameters, and scan directions with respect to scanned voltages.
B.3 Quantitative Methodology: N-way Regression

B.3.1 Pattern analysis

This paper is concerned with understanding degradation mechanisms from experimental data with the help of an algorithm to point the researcher to interesting features of the data for closer examination. When a data set is large, multi-faceted, and/or heterogeneous, it may be particularly useful to adopt this approach.

This simplest and most widely used approach for pattern recognition is direct inspection of the raw data. Arguably, this may not merit being called an “algorithm” because it is essentially a manual procedure. On the other hand, it can be done systematically with the aid of a computer.

The use of basic statistics is another way researchers in this field have tried to identify trends in data. The simplest statistical measure, correlation, is a starting point for many analyses (see, e.g., [4]). In a more recent work [61], the authors derive a novel approach for systematically examining all correlations within a J-V/C-V reliability data set (the same one used in this paper). Multivariate regression, in which one factor is modeled as a simple function of the others, is another commonly used technique [5, 84].

A slightly more computational approach to pattern recognition involves matrix decompositions. The most widely used decomposition comes from Principal Component Analysis (PCA), which identifies linear combinations of parameters that capture the most data-wide variation. This is accomplished via an eigen-decomposition of the estimated covariance matrix. PCA is used in [118], for example, to analyze the effects of various parameters on organic solar cell properties by identifying variations of each variable derived from J-V curves.

The last approach we mention is cluster analysis. In this approach, various notions of similarity may be employed to attempt to identify subsets (clusters) of parameters that are either closely related or highly dissimilar. One of the first examples of this for C-V measurements, for example, was done by Walls et al. [145, 144, 143]. While clustering is useful in some contexts, it is highly application dependent and usually requires an ad hoc definition of similarity.
B.3.2 Formulation of N-Way Partial Least Square (PLS) algorithm

Next we briefly describe the N-PLS formulation. A full mathematical description of the model is beyond the scope of this work; we refer the reader to Bro’s work [23] for details. A good overview of tensor decompositions, including but not limited to parallel factor analysis (PARAFAC), is found in [11]. Although we do not discuss it here, a more general approach to multi-way regression may be found in [18].

Before presenting more technical details, we give a simple example of data that is naturally N-way. Suppose that the temperature over time is measured at 10 different locations on the surface of large area solar cell. The data corresponding to this experiment could be cast as a two-way matrix, with times along the rows, locations along the columns, and temperature as the entries. Now suppose that, for more reliable estimates, three different sensors are used at each location and time. One can imagine adding another dimension to the matrix corresponding to the sensors, so that instead of a data matrix we have a data “cube.” This is the core idea of the N-way format, and it is a very natural way to think about data in varied experimental contexts.

In a typical linear model, the response, $y$, is assumed to be a linear combination of the variables (columns) in the data matrix, $X$. In the multi-way setting, by analogy, $y$ is taken to be a multi-linear combination of the generalized columns, or modes, of the data tensor, $\mathcal{X}$. The model is inherently flexible in that it accommodates both linear and interaction terms by default.

We consider the case of $N = 4$ (our data has this property), noting that it generalizes easily to other dimensions (the case of $N = 1, 2$ corresponds to standard two-way PLS). We assume that the input array, $\mathcal{X}$, has dimensions $I \times J \times K \times L$ and that the response vector, $y$, is of length $I$. The “loading” vectors are denoted by $t^{(f)}, u^{(f)}, v^{(f)}$ and $w^{(f)}$ and have length $I, J, K$ and $L$, respectively. Here the superscript $f$ denotes the model factor of which there are $F$ total. Subscripts $i, j, k$ and $l$ denote vector elements. When the meaning is clear, superscripts and subscripts may be suppressed for simplicity.

The N-PLS algorithm outputs $F$ sets of loading vectors, where the total number of factors, $F$, is a user-supplied parameter related to model degrees of freedom. The input and response are related mathematically
via,

\[ y = \sum_{f=1}^{F} s^{(f)} t^{(f)}, \]

\[ X_{ijkl} = \sum_{f=1}^{F} t^{(f)}_i u^{(f)}_j v^{(f)}_k w^{(f)}_l, \]

\[ t^{(f)}_i = \sum_{j=1}^{J} \sum_{k=1}^{K} \sum_{l=1}^{L} X_{ijkl} u^{(f)}_j v^{(f)}_k w^{(f)}_l. \]

The vectors \( u, v \) and \( w \) may be thought of as eigenvectors of the \( \mathcal{X} - y \) covariance tensor computed via alternating least squares. The weight, \( s^{(f)} \), is a scalar value related to the total variance of \( y \) captured by factor \( f \). Note that \( y \) is a linear combination of the \( t \) vectors, which are in turn connected to \( \mathcal{X} \) via a multi-way covariance relationship (hence, interaction terms).

The model building process is deflationary, meaning that initially a one-factor model is computed and subtracted from the input. Then another one-factor model is fit to the residual and subtracted, and so on up to \( F \) times. Hence, the first factor is common to all models, the second factor is common to all models with \( F \geq 2 \), et cetera.

Ultimately, the algorithm outputs \( F \) sets of loading vectors, each of which corresponds to some level of fit to \( y \). If one set of vectors is particularly good at fitting the response, it is inspected to see which variables it emphasizes. Thus, the loading vectors are the key to interpreting the N-PLS model. We explain the exact relationship between the loading vectors and reliability parameters in the next section.

Finally, we emphasize that expression of a multi-way array as products of vectors is, at its core, a form of dimensionality reduction. For example, suppose \( \mathcal{X} \) is experimental data with dimension \( 100 \times 10 \times 20 \), meaning it contains 20,000 measurements, and that we fit an N-PLS model with three factors (\( F = 3 \)). Because each mode is described by a vector of the same length as its dimension, the model has only \( 3 \times (100 + 10 + 20) = 390 \) total numbers, far less than the original 20,000. In general, a three-way model has a total of \( F \times (I + J + K) \), a four-way model has \( F \times (I + J + K + L) \), and so on for larger \( N \).
B.4 Results and Discussion

B.4.1 The data in N-way format

In this study, we chose $FF$ as a proxy for cell efficiency $\eta$; it is the target response, $y$, for N-PLS. While efficiency proper was explicitly measured, we wanted to use $FF$ because it is a J-V derivable - and hence physical - quantity. While $\eta$ and $FF$ are known to be closely related [4], they were also directly observed to have very high correlation in this experiment.

The complete input tensor, $X$, for N-PLS is a four-way array containing measured C-V values at all combinations of stress time, C-V parameter, DC bias, scan direction performed in the experiment (Figure B.3). It has dimensions $7 \times 4 \times 199 \times 3$. Specifically, the modes are:

- Stress time: 0, 60, 120, 300, 450, 675, 1000 hours
- CV parameter: capacitance ($C$), conductance ($G$), $W_d$, and carrier density ($N_a$)
- DC bias: -1.49V to +0.49V by increments of 0.01V
- Scan direction: forward, reverse, hysteresis

Although five devices were originally tested in the experiment, one of them behaved erratically for reasons likely caused by a manufacturing anomaly. As such, this device was removed from the sample set.

All elements of the input arrays represent averages over the devices. Averaging allowed us to emphasize degradation trends while de-emphasizing between-device variability stemming from the manufacturing process or other random sources. While this may not be desirable in all situations (for example, one might wish to distinguish degradation rate between devices), it is a reasonable step given our analysis goals.

B.4.2 Data pre-processing

An important step in many statistical techniques, no less in the N-way setting, is data pre-processing. Prior to model selection, and following some basic heuristics described in [25], we centered the time mode to ensure that ‘changes over time’, rather than ‘averages over time’, were emphasized. In addition, because the units of the raw C-V parameters differ by orders of magnitude, we re-scaled each mode to have unit
variance. In this way, we avoid the possibility of selecting variables simply because they are very large or very small compared to the rest.

B.4.3 Model selection

It is generally difficult, if not impossible, to validate a prediction-type model with only a small number of samples. In this data set - and as is very often the case in PV experiments - we are confronted with the problem of small sample size. Therefore, we deemed it necessary to apply other reasonable criteria that would allow the model to be as ‘agnostic’ as possible without over-fitting. To this end, we used two guiding principles for model selection:

(1) Parsimony: choosing the best viable model with the smallest number of factors, $F$. This is equivalent to choosing the simplest model that explains the data well.

(2) Quality of fit: a model that fits $FF$ well is desired because our aim is to correlate C-V parameters with $FF$ (and hence efficiency).

N-PLS models were built in Matlab [81] using the N-way toolbox [22]. Table B.1 shows quality of fit, in terms of both $\mathcal{X}$ and $y$, using up to three factors. Based on the above two selection principles the value of $F = 2$ was chosen. For reference, the notation for the loadings is illustrated in Figure B.4. Although the addition of a third factor slightly improves the fit, the gain is modest and so using it strains the parsimony principle. To be concrete, the measured versus fitted values of $FF$ corresponding to one- and two-factor models is shown in Figure B.5, in which points corresponding to a perfect fit would lie on the black line (measured = fitted). As discussed in next section, the two factors remarkably seem to capture distinct modes of degradation occurring over different time scales.

B.4.4 Model interpretation

As was mentioned earlier, the quality of an N-PLS model can be evaluated both in terms of fit and interpretability. Here we describe the interpretation of the model loadings, which summarize the key features in each dimension of the multi-way input, $\mathcal{X}$. Figure B.6 shows the loadings corresponding to the four different modes of $t$ (stress time), $u$ (C-V parameter), $u$ (voltage bias), and $w$ (direction of voltage
Figure B.3: A schematic of N-way data structure measured and derived from (a) C-V and (b) J-V measurements of CdS/CdTe solar cells.

Figure B.4: A schematic of selected two-component (e.g., $F = 2$) four-way PLS model. The error need not be expressible as a single term, but this is done here for brevity.
Figure B.5: Fitted versus measured $FF$ for four-way PLS. The black line represents equality between the fitted and true $FF$ values. The fit improves significantly when two factors are used rather than one, while addition of the third factor has little effect on the fit. The percentages in the legend indicate the $\%$ of $FF$ explained by that factor alone; the cumulative variance is the sum of these values.
scan). When necessary, we use subscripts to refer to specific variables in the given mode and superscripts to denote the factor. For example, \( t_3^{(2)} \) refers to the third stress time (300 hours) in the second N-PLS factor \((f = 2)\).

In interpreting the N-PLS model, we look at both:

1. The most heavily weighted variables in the individual loadings (in absolute value)
2. The relative weighting of variables between the two factors (i.e., factor maps)

The first step allows us to identify the variables that are significant within each factor, while the second will highlight differences and commonalities between the factors. It is important to do both because the factors can share some features but not others (nothing in the N-PLS formulation, as we have presented it, prevents this from happening).

Consider the time loadings in Figure B.6 (a). Notice that the factor 1 loading decreases sharply from 0 to 300 hours, after which it remains roughly constant. Factor 2, on the other hand, peaks at 120 hours and gradually decreases thereafter. Thus, we infer that factor 1 describes the rapid, short-term degradation (0-300 hours) while factor 2 accounts for a slower degradation mode (300-1000 hours) occurring over longer stress times, with some overlap in the 120-300 hour interval.

In order to see which C-V parameters correspond to the degradation modes, we examine the C-V parameter loadings in Figure B.6 (b). Apparently, the majority of the factor 1 weight falls on \( G \), indicating that the rapid initial degradation is most associated with this parameter. The long-term degradation of factor 2, meanwhile, appears to coincide almost exclusively with the \( N_a \).

Following the same line of reasoning, we see in Figure B.6 (c) that factor 1 is most strongly associated with \(+0.49\)V of voltage scan, while factor 2 has two distinguishable peaks at positive voltages, namely, at \(+0.08\)V and \(+0.18\)V, with the largest emphasis on \(+0.08\)V. In the negative voltage regime, the loading appears to pick up the noise in its associated C-V parameter, \( N_a \) (see Figure B.1 (d)). Because of this, we do not attribute any meaning to peaks in the negative bias regime.

Finally, looking at the scan direction loadings (Figure B.6 (d)), we see that the factor 1 scan loading weights the reverse, forward, and hysteresis directions in a roughly 1:2:1 ratio, while factor 2 is roughly 1:1:1.
Figure B.6: Loading vectors: (a) $t_i$ ($j = 1$ to $I = 7$), (b) $u_j$ ($j = 1$ to $J = 4$), (c) $v_k$ ($k = 1$ to $K = 199$), and (d) $w_l$ ($l = 1$ to $L = 3$), which are captured by the first two factors in the four-way PLS model. The percent values of each factor denote variances of $y$ (i.e., FF) explained by each factor. Color represents factor number ($f = 1$ to $F = 2$) as indicated in the legends. The high level of noise in the factor 2 voltage loading stems from the fact that these factors emphasize $N_a$, which is itself very noisy at negative voltages as shown in Figure B.1 (d).
From this we conclude that the N-PLS model for short stress-time degradation is best described by the forward scan measurements, while the long-term degradation is basically independent of the scan direction.

Factor maps for the model loadings are shown in Figure B.7. A factor map simply puts each of two loading vectors on an axis and shows the weights in the resulting coordinate system. (If two loading vectors are exactly equal, for example, then the plotted points would lie on a straight line with slope one passing through the origin). Such maps summarize more of the total explained variance than a single loading plot (here, two factors account for 96.5% of the variation versus 58.73% or 37.77% for one factor), as well as highlight common and disparate features of the paired loadings [154]. They are particularly convenient in our case because the model has exactly two factors.

The factor maps for time, C-V, and scan direction straightforwardly support the claims made above. One can see in Figure B.7 (a) that, indeed, factor 1 (horizontal axis) is associated with rapid change for early stress times (0-300 hours), while factor 2 changes more rapidly for later stress times (300-1000 hours). In Figure B.7 (b), we see that \( G \) and \( N_a \) are identified quite independently by the two factors, while C and \( W_d \) are not heavily weighted in either. And in Figure B.7 (d), we see again that the factors are fairly independent of the scan direction.

The most insightful factor map is that of the DC bias loadings in Figure B.7 (c). Here it is apparent that the factors are indeed associated with distinct voltage ranges. Factor 1 varies the most from 0V to +0.49V. And although factor 2 has local absolute maxima for at two positive voltage values, these changes are very small in a relative sense, a fact that may not be apparent without the factor map. Indeed, it appears that factor 2 is primarily associated with the noisy negative voltage ranges. This makes sense because the associated C-V parameter is extremely noisy for this range of biases. Of the positive biases, +0.08V is identified as the most significant for factor 2.

Based on the AC-circuit model with a capacitor in parallel, comprising two junctions of CdS/CdTe and the CdTe/back contact, Burgelman suggests that the junction capacitance is measured at low voltages \( (V < 0.3 \text{ V}) \), while contact capacitance is measured at high voltages beyond 0.49 V [27]. Based on this analysis, our results imply that \( G \) at the back contact is related to short stress-time degradation (0-300 hours), while \( N_a \) at near junction (i.e. +0.08V DC bias) is correlated with more gradual, long stress-time
degradation (300-1000 hours). For reference, a summary of the interpretation we present above appears in Table B.2.

**B.4.5 Corroboration of the model by inspection of processing conditions and raw data**

In the previous section, we drew several conclusions based on a model that incorporates all of the data simultaneously. Building and interpreting such a model circumvents the need to visually inspect subsets of the data, an approach that is inherently likely to miss global features of the data set. Once a global model is identified, however, it is reasonable to corroborate the results by doing exactly that, i.e., looking at subsets to see whether the identified trends are really present.

Analysis of the N-PLS model leads to the identification of degradation-related variables as summarized in Table B.2. The degradation mode by factor 2 is likely due to Cu-diffusion. These cells utilize a thick Cu-doped, graphite layer between the back metallization (Ag paste) and the CdTe layer. This layer serves as an “infinite” source of Cu [3]. The corresponding erfc(x) diffusion profile from this dopant layer into the CdTe has a characteristic length \( ( = 2(Dt)^{1/2} ) \) of approximately 4 microns at 60 °C and 300 hr. which is agreement with the distance between the graphite layer and the CdS/CdTe interface (i.e., the film thickness). Stress times greater than ~300 hours result in more Cu reaching the interface, increasing the density of recombination centers and reducing performance. Degradation beyond 300 hours is attributed to this mechanism.

The rapid initial degradation occurring up to ~300 hrs at 60 °C, which is captured by factor 1, is observed in cells using graphite [3]. In the current back contact procedure, the graphite layer is annealed at 260 °C for ~30 minutes. This is the optimal time specified by the manufacturer for this graphite conducting material and the subsequent heating during stress testing is one likely suspect for the initial loss in \( G \).

To validate this information, we go back to the raw data and see what, if any, connection the N-PLS model has to the original measurements. The results drawn from the model are in good agreement with trends seen in the raw data, as illustrated in Figure B.8. Figure B.8 (a) shows \( G \) at +0.49V bias versus \( FF \). The N-PLS model indicates that these quantities co-vary from 0-300 hours of stress time, and indeed this is seen to be the case in the raw data. Thereafter, \( FF \) decreases independently of \( G \) (\( G \) changes very little after 300 hours at this bias). The reverse scan direction is also seen to have high correlation with \( FF \), a
Figure B.7: Maps of the first two factors of the N-PLS model. The plots summarize degradation behaviors of cells with respect to different modes of (a) $t_i$, (b) $u_j$, (c) $v_k$ and (d) $w_l$. 
Figure B.8: Plots of raw data corresponding to the features identified by N-PLS (cf. Table B.2). (a) FF versus conductance at +0.49V DC bias. From 0 to 300 hours, both the forward/reverse scan of \( G_{\text{fwd}} \) and \( G_{\text{rev}} \) decreases with FF over time. (b) FF versus carrier density at +0.08V DC bias. From 300 to 1000 hours, \( N_a \) and FF decrease in tandem. Circles indicate the stress times identified by N-PLS. The dashed lines connect the stress hours to depict degradation trend.

fact that is not immediately obvious from N-PLS model: while reverse scan is weighted in the factor 1 scan loadings, it is less so than the forward scan. Figure B.8 (b) shows \( N_a \) versus FF values at +0.08V DC bias. As reflected in the N-PLS factor 2 time loadings, beginning at 120 hours these quantities decay together.

B.5 Conclusions and Future Work

This work demonstrates the use of N-PLS as a tool for exploratory data analysis as well for modeling correlations between multi-way C-V parameters and a performance metric, \( FF \), for the study of PV loss mechanisms of CdS/CdTe solar cells. By separately assigning multiple C-V parameters to each mode in a multi-way analysis, we effectively identified two different modes of \( FF \) loss corresponding to factors in the N-PLS model. Factor 1 describes the rapid, short-term degradation (0-300 hours) while factor 2 accounts for a slower degradation mode (300-1000 hours) occurring over longer stress times, with some overlap in the 120-300 hour interval.

In addition, the factors were associated with C-V parameters, C-V scan directions, and voltage bias. The associations detected by N-PLS generally could not be identified by visual inspection of the data without considerable effort; for larger data sets, the task becomes quite intractable. There are still challenges in fully vetting the multi-way framework. These include, for example, identifying the proper choice of pre-processing, incorporation of categorical data, and accommodation of non-linear relationships, which are partly resolved
by properly adapting advanced algorithms (for example, see [126]). Nevertheless, the multi-way approach addresses some key challenges in understanding cell degradation, namely, the integration and analysis of heterogeneous and multi-scale data for improved process control and design.

**Acknowledgement** This work was supported by the U.S. Department of Energy under Contract No. DE-AC36-08-GO28308 with the National Renewable Energy Laboratory.

Table B.1: N-PLS results for one-, two-, and three-factor models. The cumulative variance is simply the sum of the individual factor variances.

<table>
<thead>
<tr>
<th>Number of factors ($F$)</th>
<th>% $X$ variance explained by factor</th>
<th>Cumulative % $X$ variance</th>
<th>% $y$ variance explained by factor</th>
<th>Cumulative % $y$ variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25.36</td>
<td>25.36</td>
<td>58.73</td>
<td>58.73</td>
</tr>
<tr>
<td>2</td>
<td>19.97</td>
<td>36.39</td>
<td>37.77</td>
<td>96.48</td>
</tr>
<tr>
<td>3</td>
<td>6.42</td>
<td>42.81</td>
<td>3.28</td>
<td>99.76</td>
</tr>
<tr>
<td>N-PLS factor ( f )</td>
<td>( % y ) variance for factor ( f )</td>
<td>Time loading</td>
<td>C-V loading</td>
<td>DC bias loading</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------</td>
<td>--------------</td>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>1</td>
<td>58.73</td>
<td>Rapid, short stress-time degradation (0-300 hours)</td>
<td>Conductance ( G )</td>
<td>+0.49V</td>
</tr>
<tr>
<td>2</td>
<td>37.77</td>
<td>Gradual, long stress-time degradation (300-1000 hours)</td>
<td>Net acceptor density ( N_a )</td>
<td>+0.08V, and negative voltages (noisy)</td>
</tr>
</tbody>
</table>

Table B.2: Summary of N-PLS factors for the two-component model \( (F = 2) \).
Appendix c

Orthogonal projection to latent structures solution properties for chemometrics and systems biology data
C.1 Introduction

(Note: This article appeared in the Journal of Chemometrics, Volume 25, Issue 9, pages 514–525, on September 2011. It was co-authored by Peter Graf and David Astling).

Partial Least Squares (PLS), sometimes referred to as Projections to Latent Structures, is one of the most commonly used multivariate regression methods in chemometrics [56, 97, 147]. PLS is designed for applications in which the data has more independent variables than observations (the “small n, large p” regime) and where collinearity is present amongst variables. In contrast to ordinary least squares regression, which is ill-posed in the “small n, large p” regime, PLS assumes that meaningful structure in the data is low-dimensional and attempts to identify as few linear combinations of the independent variables (latent variables) as possible without sacrificing prediction quality.

Nonetheless, interpretation of PLS components can be challenging in cases where significant systematic variation is present in the measured data. Such variation can arise from a variety of sources, including temperature fluctuations, instrument and sample handling errors, and contamination of interfering substances. Even in well designed calibration studies, where the response variables are carefully controlled in order to isolate corresponding changes in the spectrum, systematic variation may still present challenges in interpreting the PLS components and may adversely affect prediction. Although one can often improve PLS prediction accuracy by introducing more components to the model, doing so has an adverse affect on model interpretability and can lead to poorer prediction quality due to over-fitting. For example, one recent study used PLS for calibration and estimation of algal lipid content from Mid-Infrared (Mid-IR) spectral data [91]. Despite the small number of changing experimental factors, a relatively large number of PLS components were needed to adequately characterize the variation in the spectral data while preserving accurate predictions of lipid concentration.

In 2002, Trygg and Wold introduced a variant of PLS, called Orthogonal PLS (O-PLS), that was designed to remove systematic variation uncorrelated with the response [136]. The idea was that doing so reduces the number of predictive components, thereby increasing interpretability of the model factors. Indeed, in calibration settings, O-PLS models are known to have similar prediction quality to PLS but often
with far fewer, more meaningful components. For nonlinear problems, the algorithm has been adapted for use with the “kernel trick” [117]. Another variant, O2-PLS, works in much the same way as O-PLS but is applied bi-directionally [137]. The data blocks are treated as predictor and response in turn to identify both $\mathbf{X}$-orthogonal variation in $\mathbf{Y}$ and $\mathbf{Y}$-orthogonal variation in $\mathbf{X}$. The O2-PLS model is constructed by applying O-PLS to the data in both directions, and then building additional linear models relating the $\mathbf{X}$- and $\mathbf{Y}$-scores.

The success of O-PLS in spectroscopy has led to its use in other fields. Recently, O2-PLS was applied in a systems biology context to integrate metabolomic and transcriptomic data [28]. The same authors have also applied it to three-way data to find relationships amongst proteomic, transcriptomic and metabolomic profiles [30]. Unfortunately, these studies never compared the O2-PLS models to analogous PLS models for the same data.

To date, theoretical descriptions of O-PLS solutions have mostly focused on their relationship to PLS, without reference to any underlying model describing the data. For instance, Kemsley and Tapp have suggested a simple method for filtering $\mathbf{X}$ using only the PLS solution [85]. This result is consistent with prior observations by Ergon [51], who has also proposed projection-based techniques to reduce the number of PLS model components [52]. Verron et. al. have also made interesting observations regarding the connections between O-PLS components and their PLS counterparts [139]. Our contribution, and the central theme of this paper, entails an analysis of O-PLS applicability to chemometric and systems biology data which, to our knowledge, has not been made in the literature. We place special emphasis on the qualitative properties of O-PLS solutions, often assuming that the data is noise-free and that $n$ is large enough for tight covariance estimates to be valid. In doing so, we explicitly outline solution properties while highlighting some of the algorithm’s strengths and limitations. In particular, following the approach set by Nadler and Coifman, we begin by analyzing O-PLS performance on spectral data obeying Beer’s law and show explicitly why the algorithm often produces informative results in this context [105, 106]. We extend this discussion to the systems biology context in which one is likely to encounter data matrices with very different properties than in calibration studies, and highlight the challenges this presents to the O-PLS algorithm.
C.1.1 Comparison of PLS and O-PLS

In order for the discussion to be reasonably self-contained, we provide a brief background on the mathematical formulation of PLS and O-PLS. Let $X$ and $Y$ be $n \times p$ and $n \times m$ matrices, respectively, where $p, m$ are the numbers of variables in each data set and $n$ is the number of observations. We assume that $Y$ is well approximated by

$$\hat{Y} = X\beta,$$

where $\beta$ is the $p \times m$ matrix of regression coefficients. While both $\hat{Y}$ and $\beta$ are understood to be sample estimates, we use a hat only on $\hat{Y}$ to distinguish it from the data, $Y$. No such clarification is needed for the matrix, $\beta$, which we always understand to be an estimator. When $n > p$ (i.e., the problem is over-determined) and $X$ is of full rank, ordinary least squares (OLS) solution, $\beta_{OLS}$, is found via the normal equations,

$$\beta_{OLS} = (X^TX)^{-1}X^TY.$$

When $n < p$ (i.e., the problem is under-determined), the matrix $X^TX$ is guaranteed to be singular and no unique $\beta_{OLS}$ satisfies (C.1).

PLS assumes that meaningful structure in the data is inherently low-dimensional, so that only a small subset of the predictor variables is necessary to predict $Y$. These subsets are traditionally called latent variables because they are, in principle, unmeasurable by themselves [97]. A succinct form of the PLS model is given by

$$X = TP^T + E$$

$$\hat{Y} = TC^T,$$

where the matrices $T, P$ and $C$ are of relatively low rank. By analogy with Principal Components Analysis (PCA), the columns of matrix $T$ are called the scores, those of $P$ the loadings (or $X$-loadings), those of $C$ the $Y$-loadings and $E$ is the residual matrix. The scores represent the coordinates of the observations with respect to the loadings, $W$, of PCA($K$), where $K = \frac{1}{n}Y^TX$ is the sample covariance matrix for mean-centered data. The $W$-loadings and $Y$-loadings are then computed so that each component is a best least squares fit. Equations (C.2) are easily rearranged to find that the PLS regression coefficient depends on both
the $X$- and $Y$- loadings via

$$\beta_{PLS} = WC^T. \quad (C.3)$$

O-PLS may be thought of as PLS combined with a pre-processing step that filters systematic variation from $X$ that is orthogonal, or statistically uncorrelated, to the $Y$ variables. The O-PLS model has the form

$$X = TW^T + T_{orth}P_{orth}^T + E$$
$$\hat{Y} = TC^T,$$

where the subscript $(\cdot)_{orth}$ denotes orthogonal components and, in general, $T$ and $C$ are not the same as those of PLS. The regression matrix for O-PLS, $\beta_{O-PLS}$, is found as in (C.3) but, given the underlying dependence of $C$ on the scores, will generally differ from the PLS regression matrix. New predictions are obtained in the standard way via (C.1) after first filtering the new samples of orthogonal variation.

For reference, we summarize the steps of O-PLS following Trygg and Wold.[137]

(1) Calculate $A$ PCA components of the $m \times p$ covariance matrix

$$PCA(K) = PCA(\frac{1}{n}Y^TX) = T_{pca}W^T, \quad (C.4)$$

where $T_{pca}$ is an $m \times A$ score matrix and $W$ is a $p \times A$ loading matrix with orthonormal columns. Recall that $A \leq m$ and that we are primarily interested in cases where $m << p$.\(^1\)

(2) Calculate the $n \times A$ predictive score matrix $T = XW$. These are coordinates of the rows of $X$ with respect to the orthonormal basis in $W$.

(3) Calculate the $n \times p$ residual matrix, $E_{xy} = X - TW^T = X - XWW^T$. This step subtracts away the projection of the rows of $X$ onto the basis in $W$.

(4) Calculate $w_{orth}$, the most significant loading of $PCA(T^T E_{xy})$.

---

\(^1\) In the event that $m = 1$, then $A = 1$ and there is no need to perform PCA. This is the case in the example in Section C.2.4.
(5) Sequentially remove structured noise, \( t_{orth} \mathbf{p}_{orth}^T \), from \( \mathbf{X} \):

\[
\begin{align*}
\mathbf{t}_{orth} &= \mathbf{Xw}_{orth} \\
\mathbf{p}_{orth}^T &= \mathbf{t}_{orth}^T \mathbf{X} / \| \mathbf{t}_{orth} \|^2 \\
\mathbf{X} &\leftarrow \mathbf{X} - \mathbf{t}_{orth} \mathbf{p}_{orth}^T.
\end{align*}
\]

(6) For additional orthogonal components, repeat Steps 4 and 5. Otherwise, repeat Step 2 using the filtered \( \mathbf{X} \) to update score matrix \( \mathbf{T} \), and go to Step 7.

(7) Construct a PLS model from \( \mathbf{Y} \) and the filtered \( \mathbf{X} \).

(8) To predict using a new sample, \( \mathbf{x}^{(n+1)} \), first filter any orthogonal variation. Then \( \hat{\mathbf{y}}^{(n+1)} = \mathbf{x}^{(n+1)} \beta_{O-PLS} \).

The authors show that the algorithm maximizes the \( L_2 \)-norm of the projection of each \( t_{orth} \) onto \( \mathbf{T} \), thereby removing as much structured noise from the \( \mathbf{Y} \)-predictive components as possible. Specifically, \( t_{orth} \) satisfies the optimization criteria

\[
\max_t (\mathbf{T}^T \mathbf{t})^2 \text{ subject to } \mathbf{Y}^T \mathbf{t} = 0. \tag{C.5}
\]

O-PLS models often have fewer components than PLS for a given accuracy. This is due to the removal of \( \mathbf{Y} \)-orthogonal components which, while potentially useless for prediction, may still describe a large fraction of the total variation in \( \mathbf{X} \). In this sense, we understand such O-PLS solutions to be more interpretable than PLS ones: if fewer components are needed for equivalent prediction quality, the corresponding latent variables should be easier to interpret.

For a description of how O-PLS relates to O2-PLS, we refer the reader to the original publication on the subject [137]. Because of the symmetric nature of the O2-PLS formulation, (first \( \mathbf{X} \) is the predictor and \( \mathbf{Y} \) the response, then vice versa), and to keep the derivations tractable, we focus only on O-PLS. We do so with the understanding that all arguments also apply in reverse, with \( \mathbf{X} \) considered the response matrix and \( \mathbf{Y} \) the predictor matrix.
C.2 O-PLS solution properties for data obeying Beer’s law

C.2.1 Beer’s law and calibration data

The original O-PLS literature considered application of the algorithm to spectroscopic data [136]. In this section, we show explicitly why O-PLS results may be easier to interpret than PLS results for Mid-IR calibration data, but also how certain undesirable solution features can arise depending on various properties of the data. By imposing different restrictions on the input data, we can analyze directly the strengths and limitations of the algorithm.

The theoretical foundation for Mid-IR calibration is Beer’s law, which states that the spectral data is a linear mixture of the pure constituent profiles weighted by their concentrations. In an appropriately chosen system of units, the law may be written as a sum of outer products of all constituent concentration and pure profile vectors,

\[ X = \sum_{i=1}^{s} y_i z_i^T. \]  

(C.6)

Here \( X \) is the \( n \times p \) matrix of spectral data, \( s \) is the total number of constituents in the sample, \( y_i \) is an \( n \)-vector of concentrations, and \( z_i \) is the \( p \)-dimensional pure constituent profile (extinction coefficients) of species \( i \). It is most often the case in calibration studies that only a relatively small number \( j \) of the \( s \) concentrations are of interest, making it convenient to separate the expression (C.6) into two sums, the first corresponding to analytes and the second to background constituents whose properties are, in general, unknown. We assume the indices have been chosen so that the first \( j \) components \( i = 1, ..., j \) are the analytes and the next \( k = s - j \) components \( i = j + 1, j + 2, ..., s \) are the background constituents. We continue to use the symbols \( y \) and \( z \) to denote analyte concentrations and spectral profiles, but choose \( u \) and \( v \) to denote the analogous background properties. Beer’s law may then be expressed as

\[ X = \sum_{i=1}^{j} y_i z_i^T + \sum_{i=1}^{k} u_i v_i^T, \]  

(C.7)

with \( s = j + k \). To simplify the notation one step further, we matricize the summations in (C.7) to get

\[ X = YZ^T + UV^T, \]  

(C.8)
where $Y$ ($n \times j$) and $Z$ ($p \times j$) contain the $\{y_i\}_{i=1}^j$ and the $\{z_i\}_{i=1}^j$ in their columns, respectively, and similarly for $U$ ($n \times k$), $V$ ($p \times k$), $\{u_i\}_{i=1}^k$ and $\{v_i\}_{i=1}^k$.

Some authors have employed a probabilistic model of the input data, where the measured concentrations are samples from a potentially noisy underlying distribution [105, 106]. Although we recognize the theoretical value of a probabilistic approach, we feel that most researchers actively using PLS and O-PLS/O2-PLS will be more familiar with a discrete formulation. That said, statistical estimates for finite training data may contain large variance, especially when $n$ is small. Because we are primarily interested in the qualitative properties of O-PLS solutions, we will often assume that $n$ is large enough for certain covariance estimates (e.g., $\text{Cov}(Y, U) = 0$) to be valid. The only exception to this is in the following Section C.2.3, in which we examine how the algorithm handles noisy data as a function of $n$ and the noise strength.

In the following, we assume mean centering of the data, and denote the covariance of two matrices $A$ and $B$ by $\text{Cov}(A, B) = \frac{1}{n}A^TB$. With some abuse of notation, we define analogous quantities between matrices and vectors, and vectors and vectors. We also note that variance and covariance are related in the usual way, so that for any vector $a$, $\text{Var}(a) = \text{Cov}(a, a)$.

C.2.2 Uncorrelated concentrations and non-overlapping pure profiles

We first consider the simplest possible assumptions about data obeying Beer’s law (C.8). We show that PLS and O-PLS solutions are identical, in particular because O-PLS does not find $Y$—orthogonal variation. The assumptions are then relaxed to obtain more general results. Unless otherwise stated, we take the data to be noise-free and $n$ to be “large enough” to justify ignoring high order terms in the covariance estimates (though we do briefly describe how O-PLS handle noise for small $n$ in Section C.2.3).

Suppose the system has the following properties:

**Assumption 1:** uncorrelated concentrations, $\text{Cov}(Y, U) = 0$,

**Assumption 2:** non-overlapping pure profiles, $Z^TV = 0$.

Physically, these assumptions imply that the analyte concentrations change completely independently of the background across all samples, and that their pure profiles share no spectral peaks with the background.
While this situation may be unlikely in real experiments, it is a useful starting point for the analysis because of its simplicity.

First, we note that both PLS and O-PLS covariance loadings, $\mathbf{W}$, are linear combinations of the pure analyte profiles. To see this, recall that the first step of both algorithms computes the $\mathbf{XY}$-covariance matrix which, by Assumption 1, reduces to

$$
\mathbf{K} = \frac{1}{n} \mathbf{Y}^T \mathbf{X} = \frac{1}{n} \mathbf{Y}^T \mathbf{YZ}^T = \text{Cov}(\mathbf{Y}) \mathbf{Z}^T. 
$$

(C.9)

From this it follows that the columns of the $p \times A$ loading weight matrix, $\mathbf{W}$, of PCA($\mathbf{K}$) must be linear combinations of the pure profiles and hence lie in the subspace $\text{Span}\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_j\}$, having dimension $d_j$. Then, by Assumption 2, the background pure profiles must be orthogonal to all of the loading weights, $\mathbf{V}^T \mathbf{W} = \mathbf{0}$.

Second, both PLS and O-PLS find identical score matrices whose columns are linear combinations of the analyte concentrations,

$$
\mathbf{T} = \mathbf{XW} = \mathbf{YZ}^T \mathbf{W} = \mathbf{YA}_T, 
$$

(C.10)

weighted by the spectral projection matrix $\mathbf{A}_T = \mathbf{Z}^T \mathbf{W}$. This is desirable from a prediction standpoint because the background constituents are assumed to be uncorrelated with the analytes. Note that when there is only one analyte ($j = 1$), Equation (C.10) implies that the first score is exactly proportional to the analyte concentration.

Under the above conditions, the first orthogonal component of O-PLS is zero. To see this directly, first compute the $\mathbf{X}$-residual,

$$
\mathbf{E}_{xy} = \mathbf{X} - \mathbf{TW}^T = (\mathbf{YZ}^T + \mathbf{UV}^T) - \mathbf{YZ}^T \mathbf{WW}^T = \mathbf{UV}^T,
$$

where we have used the fact that $\mathbf{Z}^T = \mathbf{Z}^T \mathbf{WW}^T$ because $\mathbf{W}$ contains a basis for the columns of $\mathbf{Z}$. This and Assumption 1 yield the result directly,

$$
\frac{1}{n} \mathbf{E}_{xy}^T \mathbf{T} = \mathbf{VCov}(\mathbf{U}, \mathbf{Y}) \mathbf{A}_T = \mathbf{0}. 
$$

(C.11)

No $\mathbf{Y}$-orthogonal variation is found and both PLS and O-PLS produce exactly the same predictive model.
Omitting the simple derivations for brevity, we note that for a single-response under these conditions, the O-PLS solution has the properties:

\[ t \propto y, \quad p, \beta_{O-PLS} \propto z, \quad w_{orth} = 0. \]

**C.2.3 Relation to Gaussian noise**

We note that when \( n \) is small, the covariance estimate in Assumption 1 will be \( O(1/\sqrt{n}) \) rather than zero, with a constant that depends on \( Y \) and \( U \). A careful look at the steps of the algorithm reveals that this can lead to correction terms of \( O(1/\sqrt{n}) \) in (C.11). Any implementation of O-PLS will have an orthogonality tolerance to determine whether the matrix in (C.11) contains significant components. So long as the error in the covariance estimate stays below this tolerance, the results of the previous paragraphs remain unchanged.

Regarding the algorithm’s handling of noise, the authors of O-PLS claim that “results from initial studies with O-PLS do not show any degradation of results compared to non-treated data” [136]. Indeed, in the limit of large \( n \) when the covariance estimates are precise, noise is uncorrelated with the analyte concentrations and O-PLS classifies it as the residual. For small \( n \), however, it is possible that O-PLS will identify noise as being orthogonal to \( Y \).

We make these statements more precise and illustrate with a simple example. Suppose that the data is modeled by Beer’s law (C.8) with noise,

\[ X = YZ^T + UV^T + \sigma \xi, \]

where \( \sigma \) is a nonnegative parameter characterizing noise strength, and \( \xi \) is a matrix whose elements are i.i.d. samples from \( \mathcal{N}(0, 1) \). For large enough \( n \), the Assumptions 1 and 2 remain valid and the above results still hold. In particular, no orthogonal variation is identified by O-PLS, and the O-PLS and PLS models are the same.

For small \( n \), however, we must instead assume that \( \text{Cov}(Y, \xi) = O(1/\sqrt{n}) \). While a detailed analysis of the solution properties’ dependence on \( \sigma, n, \) and \( p \) are beyond the scope of this paper, we can demonstrate that the following properties are true for a single-component system with Gaussian noise:
(1) \( \max |w_{orth}| = O(\sigma^2/n) \). This quantity is directly related to the tolerance used by O-PLS to identify orthogonal variation.

(2) Filtering of noise by O-PLS can lead to lower residuals, but at the cost of over-fitting.

We omit a detailed proof of these statements and instead present a numerical example to support the claims.

C.1 shows O-PLS solution properties for a single-component system with noise. The noise parameter \( \sigma \) is allowed to vary from 0 to 0.04, where \( \sigma = 0 \) corresponds to a perfectly clean signal and \( \sigma = 0.04 \) to a highly corrupted one (left panel). The center panel provides numerical evidence for the claim that \( \max |w_{orth}| = O(\sigma^2/n) \). Note that on a log-log scale, and for all fixed \( n \), the dependence of \( \max |w_{orth}| \) on sigma is approximately linear with slope two (it is quadratic in \( \sigma \)). On the other hand, for all fixed \( \sigma \), the spacing between the lines is approximately constant. Since \( n = 2^l \) for \( l = 1, \ldots, 6 \), this fact implies that \( \max |w_{orth}| \) is also proportional to \( 1/n \).

Recall that magnitude of \( w_{orth} \) determines whether O-PLS identifies orthogonal variation (Step 4 of the O-PLS pseudocode). Hence, as this quantity increases, we would expect O-PLS to identify more orthogonal components. This is illustrated in the right panel of C.1, which shows the average number of orthogonal components identified as a function of \( \sigma \) for \( n = 64 \). In region A, corresponding to small \( \sigma \), no orthogonal variation is found. In region B, either zero or one orthogonal components are identified, depending on the specific instance of the noise. As \( \sigma \) increases further (region C), O-PLS consistently identifies one orthogonal component, and so on for regions D and E. Although not shown for clarity, similar trends are apparent for other values of \( n \), with the cutoff for each region shifted further to the left (smaller \( \sigma \)) for smaller \( n \). Although the residual for O-PLS decreases as the number of orthogonal components increases, this may be viewed as overfitting since the algorithm is building a model of the noise that will be used to filter future samples.
Figure C.1: O-PLS model dependence on noise strength $\sigma$ and sample size $n$ for a simple system. The spectral data are of the form $X = yz^T + \sigma \Xi$, where $y$ is mean centered and of unit variance and $z$ is the pure profile with $\max |z| = 2$. The noise is comprised of a scalar $\sigma$ and matrix $\xi$ whose columns are random samples from a Gaussian distribution with mean zero and unit variance. Left: Dependence of sample spectra on noise parameter $\sigma$. Center: The max norm of $w_{\text{orth}}$ is of order $O(\sigma^2/n)$. The norm of this vector determines whether or not O-PLS identifies orthogonal components. Right: PLS and O-PLS $\log_{10}$ RMSEP as a function of $\sigma$ over 50 trials for $n = 64$. Below the O-PLS threshold for $w_{\text{orth}}$ (interval A), no orthogonal components are found and O-PLS prediction is equivalent to PLS. As the magnitude of $w_{\text{orth}}$ increases, O-PLS identifies one (interval C) and then two (interval E) orthogonal components. Intervals B and D are transition regions where the average number of orthogonal components falls between the adjacent values. Similar trends are apparent for other values of $n$, with the main difference being that the cutoff is shifted towards smaller $\sigma$ for smaller $n$. 
C.2.4 Uncorrelated concentrations and overlapping pure profiles

The assumption of uncorrelated concentrations may be valid in calibration experiments where the known constituents are added directly to an otherwise homogeneous sample set (i.e., in “spike-in” calibration studies[91]). While the concentrations of the analytes vary, those of the unknown background constituents should remain roughly the same from sample to sample, resulting in zero or near-zero sample correlations between analyte and background concentrations. On the other hand, the second assumption of non-overlapping pure profiles is often invalid because the spectra of many constituents share peaks.

Therefore, we next examine PLS and O-PLS models under only Assumption 1 (Cov(Y, U) = 0), allowing for the possibility of spectral overlap (Z TV ≠ 0). We find that O-PLS interpretability (corresponding to a smaller number of components) is more robust against increases in spectral overlap and concentration covariance than PLS. In particular, we will show that O-PLS solution components have the following properties:

1. \( w_{orth}^T Z = 0 \), i.e. the orthogonal loading is orthogonal to the pure profiles.
2. \( t_{orth} \) is a linear combination of the background concentrations, \( \{ u_i \}_{i=1}^k \).
3. Removing the orthogonal component, \( p_{orth} = t_{orth} \), removes only some linear combination of the background profiles.

Furthermore, for a single-response system, we derive the following properties:

\[
\begin{align*}
  t_1 & \propto y, & p &= z, & \beta_{O-PLS} &= \frac{\tilde{z}}{||z||}, \\
  t_{orth} &= u, & p_{orth} &= v, & X_1 &= yz^T
\end{align*}
\]

where the inverted hat \( \tilde{\cdot} \) denotes a unit vector.

To motivate the discussion, we first present simple numerical results for a three-component system in C.2. The system has one analyte and two background constituents with concentrations \( y, u_1, \) and \( u_2 \) and spectral profiles \( z, v_1, \) and \( v_2 \), respectively. Each column of the figure highlights qualitative properties of PLS and O-PLS solutions for a different degree of overlap of the pure profiles (overlap increases from left to right), on the assumption that all concentrations are uncorrelated (not shown). In this example, the single
O-PLS component has a score vector collinear with $\mathbf{y}$ and a loading proportional to the pure profile. PLS, on the other hand, produces a three-component model with the same prediction accuracy as O-PLS, but whose loadings are more difficult to interpret. We outline the reasons for this below, occasionally referring to the figure to illustrate the key points.

The covariance matrix and its PCA loadings are the same as before for both PLS and O-PLS (C.9). However, the PLS score matrix is given by

$$\mathbf{T} = \mathbf{XW} = \mathbf{YB}_T + \mathbf{UC}_T,$$

where we have defined $\mathbf{B}_T = \mathbf{Z}^T \mathbf{W}$ and $\mathbf{C}_T = \mathbf{V}^T \mathbf{W}$. Written in this way, Equation (C.12) emphasizes how PLS scores may contain contributions from potentially $\mathbf{Y}$-orthogonal background constituents. The extent to which this happens in general depends on:

- the variances of analyte and background concentrations. Because the variance of each individual concentration vector is proportional to its norm, columns of $\mathbf{Y}$ and $\mathbf{U}$ in (C.12) that have large variances will significantly affect the geometric direction and magnitude of the columns of $\mathbf{T}$.

- the parameter $a_T \equiv ||\mathbf{C}_T||/(||\mathbf{C}_T|| + ||\mathbf{B}_T||)$, which we call the profile overlap ratio because it characterizes the extent to which the pure profiles of background constituents overlap with those of the calibration ones. Note that the ratio is zero when the background profiles are all orthogonal to the loading weights, $\mathbf{W}$, and approaches 1 as the background profiles lie increasingly in the subspace spanned by the basis in $\mathbf{W}$.

If either $\text{Var}(\mathbf{U})$ or the overlap ratio is close to zero, the second term in (C.12) will not drastically contribute to the scores. In any other case, however, the second term may be non-negligible. Unfortunately, because the PLS score matrix contains contributions from $\mathbf{U}$, which is assumed to be orthogonal to $\mathbf{Y}$, the prediction accuracy of individual components is necessarily degraded. This type of variation, although not caused by measurement or sampling error, is noise with respect to prediction of the analytes and is exactly the type of variation O-PLS aims to characterize and remove (cf., third row of C.2).
Concentration covariance and spectral overlap also affect PLS loadings given by

$$\mathbf{P}^T = (\text{Left}^T \mathbf{T}^T \mathbf{T})^{-1} \mathbf{T}^T \mathbf{X}. \quad \text{(C.13)}$$

While a closed-form expression for the inverse, $(\mathbf{T}^T \mathbf{T})^{-1}$, is intractable, we can rewrite the product $\mathbf{T}^T \mathbf{X}$ as

$$\mathbf{T}^T \mathbf{X} = \mathbf{B}^T \mathbf{Y}^T \mathbf{Y} \mathbf{Z}^T + \mathbf{C}^T \mathbf{U}^T \mathbf{U} \mathbf{V}^T = \mathbf{B}_p \mathbf{Z}^T + \mathbf{C}_p \mathbf{V}^T.$$

The $A \times j$ and $A \times k$ matrices $\mathbf{B}_p = n \mathbf{B}_T \text{Cov} (\mathbf{Y})$ and $\mathbf{C}_p = n \mathbf{C}_T \text{Cov} (\mathbf{U})$ are defined to simplify the expression. Written in this way, it is clear that the loadings depend on both $\mathbf{Z}$ and $\mathbf{V}$. Hence, like the scores, PLS loadings may contain contributions from the unknown constituents, the strength of which depends in a complex way on spectral overlap and auto-covariance of the calibration and background constituent groups (cf., fourth row of C.2). Contrast this with the loadings described in Section C.2.2 that depend only on the analyte profiles, $\mathbf{Z}$.

Next, we show that O-PLS scores and loadings differ significantly from their PLS counterparts, resulting in fewer predictive components and better interpretability of the loadings. First, note that O-PLS will initially compute the same scores as PLS, $\mathbf{T} = \mathbf{YB}_T + \mathbf{UC}_T$. It is straightforward to show that $\mathbf{w}_{\text{orth}}$ is orthogonal to all of the pure profiles, $\mathbf{w}_{\text{orth}}^T \mathbf{Z} = 0$. Hence, up to normalization, the first orthogonal score vector has the form

$$\mathbf{t}_{\text{orth}} = \mathbf{Xw}_{\text{orth}} = \mathbf{UV}^T \mathbf{w}_{\text{orth}}. \quad \text{(C.14)}$$

The columns of $\mathbf{t}_{\text{orth}}$ are linear combinations of the $\{\mathbf{u}_i\}_{i=1}^k$ only, meaning that each orthogonal component corresponds only to variation from the background. This is desirable because, by assumption, the background is uncorrelated with $\mathbf{Y}$ and should not contribute to the prediction model. In C.2, the single O-PLS predictive score is compared with the first PLS score in the third row. One can infer from these plots that the two orthogonal scores removed by O-PLS are some linear combination of the background concentrations because the only remaining score is perfectly collinear with $\mathbf{y}$.

Analysis of the O-PLS loadings is not as simple as for PLS because they are computed after iteratively filtering $\mathbf{X}$. However, it is useful to examine the orthogonal loadings which, up to normalization, have the form

$$\mathbf{p}_{\text{orth}} = \mathbf{t}_{\text{orth}}^T \mathbf{X} = n \mathbf{w}_{\text{orth}}^T \mathbf{V} \text{Cov} (\mathbf{U}) \mathbf{V}^T. \quad \text{(C.15)}$$
Clearly, each orthogonal component, $t_{orth}p_{orth}^T$, subtracts some linear combination of the pure background profiles from $X$. Again, the degree to which this happens depends in a complex way on the auto-covariance of the background concentrations and the degree of pure profile overlap. For our simple example, the O-PLS loading contains no contribution from background profiles (C.2).

Closed-form analysis of O-PLS solutions for the most general case is beyond the scope of this paper. For simple systems with only a single response, however, it is easy to verify that O-PLS solutions have the above properties.

Consider a two-component system,

$$X = yz^T + uv^T,$$

corresponding to (C.8) with $j = k = 1$, in which concentrations are uncorrelated ($\text{Cov}(y, u) = 0$) and the pure profiles overlap ($z^Tv \neq 0$). We omit the explicit calculation of the PLS loadings, simply noting that they are some linear combinations of both $z$ and $v$ (cf., C.2). In relatively few lines, we can show that O-PLS exactly separates the analyte from the background.

We first note that the first normalized loading weight is proportional to the pure analyte profile, $w_1 = z/||z||$. This follows from the fact that the covariance vector,

$$k = \frac{1}{n}y^T(yz^T + uv^T) = \text{Var}(y)z^T,$$

analogous to $K$ in (C.4), is collinear with $z$ and proportional to $w_1$ by construction. The initial score contains a $u$-component,

$$t_1 = Xw_1 = (yz^T + uv^T)z/||z|| = \alpha y + \beta u,$$

where for convenience we define $\alpha = ||z||$ and $\beta = z^Tv/||z||$. O-PLS next computes the residual,

$$E_{xy} = X - t_1w_1^T$$

$$= yz^T + uv^T - (\alpha y + \beta u)\frac{z^T}{\alpha}$$

$$= u \left( v^T - \frac{\beta}{\alpha}z^T \right).$$
and orthogonal loadings weight,
\[
\tilde{w}_{\text{orth}}^T = t_1^T \mathbf{E}_{xy} = (\alpha y + \beta u)^T u \left( v^T - \frac{\beta}{\alpha} z^T \right) \\
= n \beta \text{Var}(u) \left( v^T - \frac{\beta}{\alpha} z^T \right),
\]
where the tilde simply indicates that the vector is not yet normalized. Note that \( \tilde{w}_{\text{orth}} \) is proportional to the \( z \)-orthogonal projection of \( v \) because
\[
v - \frac{\beta}{\alpha} z = v - \frac{v^T z}{||z||^2} z = v - (v^T \hat{z}) \hat{z}
\]
where \( \hat{z} \) is the unit vector in the direction of \( z \).

Now let \( w_{\text{orth}} = \tilde{w}_{\text{orth}}/||\tilde{w}_{\text{orth}}|| \), where \( ||\tilde{w}_{\text{orth}}|| = n \beta \text{Var}(u) (||v||^2 - ||v^T \hat{z}||^2) \). The fact that \( w_{\text{orth}} \) is orthogonal to \( z \) implies that the orthogonal score reduces to exactly \( u \) via
\[
t_{\text{orth}} = X w_{\text{orth}} = u \frac{n \beta \text{Var}(u)(||v||^2 - ||v^T \hat{z}||^2)}{||\tilde{w}_{\text{orth}}||} = u.
\]
This leads to the first and only orthogonal loading being exactly equal to the background profile by
\[
p_{\text{orth}}^T = t_{\text{orth}} X/||t_{\text{orth}}||^2 = u^T u v^T/||u||^2 = v^T.
\]

It follows immediately that the background component is removed because \( t_{\text{orth}} p_{\text{orth}}^T = u v^T \), and that the filtered data is exactly equal to the \( y \)-component, \( X_1 = y z^T \). O-PLS recomputes the first score,
\[
t_1 = X_1 w_1 = \frac{y z^T z}{||z||} = ||z|| y,
\]
with corresponding (scalar) \( y \)-loading,
\[
c_1 = \frac{t_1^T y}{||t_1||^2} = \frac{||z|| y^T y}{||z||^2 y^T y} = \frac{1}{||z||}
\]
and regression vector,
\[
\beta_{\text{O-PLS}} = c_1 w_1 = \frac{z}{||z||^2} = \frac{\hat{z}}{||z||},
\]
which is also proportional to the loading \( p \). Although not derived here, we note that PLS also arrives at the same regression coefficient.

In summary, O-PLS filters exactly the \( u \)-component of the data resulting in an easily interpreted, single-component model. While PLS arrives at the same regression coefficient, each of the three scores and
loadings depend significantly on the degree of overlap of the pure profiles. This example contains only two constituents for simplicity, but analogous results holds for the example in C.2 and should generalize easily to more complex systems.

**Relation to baseline drift**

A common type of $\mathbf{Y}$-orthogonal measurement error is baseline drift, whereby the mean of the sample spectra appears to vary for each sample [136]. We discuss in Section C.3 how it is, in fact, possible for O-PLS not to filter this type of variation for certain types of data matrices, though the conditions under which this can happen are not likely to be found in calibration data. We ignore this possibility for the time being and show explicitly, for calibration data at least, why O-PLS identifies the drift as $\mathbf{Y}$-orthogonal.

A useful way to think of baseline drift is as noise uncorrelated with the calibration concentrations but whose loading is a non-zero constant. As such, the noise “pure profile” may be thought of as overlapping with the constituent pure profiles and identified as $\mathbf{Y}$-orthogonal. To see this in a simple case, replace the background component in the previous example with a one of the form $\mathbf{\sigma}\xi^T$, where $\mathbf{\sigma}$ is a normally distributed $n$-vector with mean zero and unit variance (uncorrelated with $\mathbf{y}$) and $\xi = \xi_1$ is a constant “baseline” vector, equal to a scalar, $\xi$, times the one-vector, $\mathbf{1}$. Thus the drift plays the role of the background in the example and is shown to be filtered accordingly.

**C.2.5 PLS prediction optimality for noise-free data and relation to O-PLS**

In this section we provide justification for the observation that, in practice, O-PLS prediction is very similar to that of PLS. This means that, under the conditions in Section C.2.4, O-PLS may have fewer components while being just as predictive as PLS. We begin by citing a relevant result about PLS prediction optimality by Nadler and Coifman [105]. We then show that PLS regression in the sense of (C.1) is identical whether one uses $\mathbf{X}$ or a filtered $\mathbf{X}$, provided that the filtering is done in a certain way. In particular, PLS prediction does not depend on the component of $\mathbf{z}$ that is orthogonal to the net analyte signal vector. This suggests a mechanism by which O-PLS filters the data without impairing prediction.

Following the analysis by Nadler and Coifman, consider a single-response, noise-free system obeying
Beer’s law (Equation C.8 with \( j = 1 \) and \( k \geq 1 \)) [105]. We assume that the \((k+1) \times (k+1)\) covariance matrix of \(\{y, u_1, \ldots, u_k\}\) is of full rank, and that \(z \notin \text{Span}\{v_1, v_2, \ldots, v_k\}\). The analysis also depends on the notion of a \textit{net analyte signal} (NAS) vector of the response, \(y\), which is commonly defined as the component of the analyte profile that is orthogonal to the background profiles [24, 94],

\[
z_{NAS}(y) = z - \sum_{i=1}^{d_k} (z^T \hat{v}_i) \hat{v}_i.
\]

Here the set \(\{\hat{v}_i\}_{i=1}^{d_k}\) is an orthonormal basis for \(\text{Span}\{v_1, v_2, \ldots, v_k\}\). Note that, by construction, the NAS is orthogonal to all of the background profiles,

\[
z_{NAS}^T V = 0. \tag{C.16}
\]

Nadler and Coifman show that, for an infinite training set with no noise, the following two results hold:

1. the RMSEP of a PLS model with at most \(k+1\) components is zero, and

2. the corresponding regression vector \(\beta_{PLS}\) is equal to a constant \(b\) times the net analyte signal vector,

\[i.e., \beta_{PLS} = b z_{NAS}.
\]

Combining these two results, we find

\[
\hat{y} = X\beta_{PLS} = (yz^T + UV^T)\beta_{PLS} = yz^T b z_{NAS}.
\tag{C.17}
\]

by Equation (C.16) and Result 2. This shows that exact prediction does not depend on the \(k\) background constituents. Under the stated conditions, any filtering procedure that does not alter the \(y\)-component of the data does not influence prediction optimality of PLS regression.

In fact, letting \(z_{NAS}^\perp = z - z_{NAS}\), we continue the derivation in (C.17) to find

\[
X\beta_{PLS} = y(z_{NAS} + z_{NAS}^\perp)^T \beta_{PLS} = (yz_{NAS}^T) \beta_{PLS}.
\]

From this we conclude that any algorithm filtering \(X\) of its NAS-orthogonal components will not affect prediction optimality under the above conditions.
These observations do not make reference to an explicit O-PLS model for the most general case of correlated concentrations and overlapping pure profiles, but they do suggest an explanation for the similarity of PLS and O-PLS predictions. As long as filtering preserves the $y$-component completely, or the NAS-collinear part of $y$, regression on the filtered data yields identical prediction. And this “prediction agnostic” filtering is exactly the type that O-PLS is expected to perform. The question of how exactly O-PLS predictions are affected by the presence of noise for finite training data remains open. Nadler and Coifman derive several interesting PLS results for noisy, finite training data which presumably could be extended to O-PLS [106].

C.2.6 General calibration

For the most general calibration data, it may not be appropriate to assume that analyte and background concentrations are uncorrelated. Here we remove that assumption ($\text{Cov}(Y, U) \neq 0$) and place no constraints on the pure profile relationships ($Z^T V \neq 0$). We demonstrate that filtering the variation can adversely affect interpretability of the resulting model. Furthermore, we show that it is likely in this case that no $Y$-orthogonal variation exists at all although, intuitively, one might expect it to exist.

A simple example is sufficient to illustrate how filtering of $Y$-orthogonal variation in the most general case can adversely affect interpretation of the O-PLS model. Consider a three-component system with one analyte characterized by concentration $y$ and pure profile $z$, and two background constituents characterized by concentrations $u_1, u_2$ and spectral profiles $v_1, v_2$ (C.3). We assume that $y$ is correlated with $u_1$ but not with $u_2$, and that $u_1$ and $u_2$ are only partially correlated. In addition, all three pure profiles overlap in the middle region of the spectrum, corresponding to the case $a_T = 0.316$ in C.2. Intuitively, one might expect it to be beneficial to remove one orthogonal component because $u_2$ is uncorrelated with $y$. Indeed, the removal of one orthogonal component results in a two-component model, where each score is more collinear with $y$ than any of the three PLS scores. One can see in the figure that the orthogonal loading positively weights the background peaks. However, removing this component degrades the interpretation of the model because the peaks of $v_1$ are positively weighted in the regression vector. In contrast, the PLS regression vector shows positive weights for peaks in the analyte profile, $z$, and negative weights for the background peaks.
Figure C.2: Comparison of PLS and O-PLS models for different levels of pure profile overlap. **First row:** Pure constituent profiles with various degrees of overlap. **Second row:** Three representative spectra. **Third row:** The first PLS and O-PLS scores (normalized for ease of visualization) are plotted against $y$ for each value of $a_T$. **Fourth row:** Interpretability of PLS loadings is not straightforward (dashed lines) and changes significantly with $a_T$, while the single O-PLS loading remains proportional to the pure profile $z$ independent of degree of overlap.
Intuitively, the reason for poor interpretability of the regression coefficient is that O-PLS removes the $y$-orthogonal component of both background peaks simultaneously, while only one of the background components is truly uncorrelated with the analyte. We find by inspection that the orthogonal loading $p_{orth}$ is approximately equal to the first loading of the truly $y$-orthogonal component of $X$, denoted $X^\perp$,

$$X^\perp \approx u_1^\perp v_1^T + u_2^\perp v_2^T,$$

where $u_1^\perp$ and $u_2^\perp$ are the $y$-orthogonal components of $u_1$ and $u_2$. Hence, the filtered data is approximately equal to

$$X_1 \approx X - X^\perp = y(z^T + \gamma v_1^T).$$

where $\gamma = \frac{\text{Cov}(u_1, y)}{\text{Var}(y)}$. It is as if the system now contains just the analyte, but with a modified profile, $z^T + \gamma v_1^T$. This is apparent in the figure, where one can see that the regression coefficient has significant contributions from $v_1$, and essentially no interference from $v_2$. While it is true that the peaks related to $y$ and $u_1$ covary in the spectral data on account of correlated concentrations, they are not generally useful in predicting $y$.

Under the above conditions, it is also possible that no identifiable $Y$-orthogonal variation exists. Consider the general covariance matrix for both PLS and O-PLS,

$$K = \frac{1}{n} (Y^T Y Z^T + Y^T U V^T) = \text{Cov}(Y) Z^T + \text{Cov}(Y, U) V^T. \quad (C.18)$$

Recall that, by Equation (C.4), the loading weights, $W$, are the PCA loadings of $K$. From (C.18), it is clear that these loading weights have contributions from the background constituent profiles, the significance of which depends directly on analyte-background covariance. If $K$ is of full rank, say with rank $r_k$, and the subspace spanned by the pure profiles (both calibration and background) is of dimension less than or equal to $r_k$, then $W$ will contain an orthonormal basis for the row space of $X$. It follows that $E_{xy} = 0$, and O-PLS will find no orthogonal components. We explicitly outline properties of the data matrices that can lead to this situation in Section C.3.1.
Figure C.3: Comparison of PLS and O-PLS solution properties for system with overlapping pure profiles and concentrations (not shown) satisfying $\text{Cov}(y, u_1) = 0.45$, $\text{Cov}(y, u_2) = 0$, and $\text{Cov}(u_1, u_2) = 0.04$. Note that $u_2$ is totally uncorrelated with $y$ and has low correlation with $u_1$, but its pure spectrum overlaps with both. Top left: Pure analyte profile ($z$) and background pure profiles ($v_1$ and $v_2$). Top right: First O-PLS orthogonal loading. Bottom left: Comparison of fit for first O-PLS and PLS components. Bottom right: Comparison of O-PLS and PLS regression coefficients. Despite the orthogonality of $u_2$ and $y$, removal of one orthogonal component leads to a misleading regression coefficient that positively weights the peaks of $v_1$ which, in general, are not useful for predicting $y$. 
C.3 O-PLS for systems biology

C.3.1 Y-orthogonal variation may not exist

So far we have considered the performance of O-PLS on calibration data modeled by Beer’s law. The benefit of having such a model is that it allows explicit analysis of solution properties and prediction capability. Although O-PLS interpretability can suffer when analyte and background concentrations are highly correlated, the above analysis shows that it is otherwise quite effective at reducing the required number of predictive components.

Perhaps not surprisingly, in light of the success of O-PLS in chemometrics applications, recent attempts have been made to apply O-PLS to large two-block data sets in the context of systems biology. For large matrices, however, the data may not contain the type of orthogonal variation that the algorithm is designed to filter. We outline some fairly broad conditions that lead to this situation. By broad we mean conditions that may be commonly encountered when the number of variables in \( X \) and \( Y \) is large relative to the number of samples, as might be the case in studies involving microarray or metabolomic data with anywhere from hundreds to tens of thousands of variables. Barring successful filtering of structured \( Y \)-orthogonal variation, the model produced by O-PLS is identical to a PLS model of the original data. We note that, in contrast to the previous section, the analysis of this section does not depend on assumptions of large \( n \) or the absence of noise.

Before presenting technical details, we take a general view of how O-PLS may encounter problems when both data matrices have a large number of variables. Suppose both \( X \) and \( Y \) contain far more variables than observations \((m, p >> n)\). The purpose of both PLS and O-PLS is to find linear combinations of the independent \( X \)-variables that simultaneously describe variation in \( X \) and are significant for the prediction of \( Y \). Recall that O-PLS filters portions of the scores that are orthogonal to the columns of \( Y \), while PLS scores may have \( Y \)-orthogonal components. However, when \( Y \) is “large enough” and “enough” of the response variables are linearly independent, the subspace orthogonal to the columns of \( Y \) is empty. In other words, because each column of \( Y \) is a vector in the vector space \( \mathbb{R}^n \) spanned by an \( n \)-dimensional basis set, if enough of the columns are linearly independent then all \( n \) of those basis vectors will be needed to construct the best
possible predictive scores, leaving no basis vectors to describe a possibly \( \textbf{Y} \)-orthogonal subspace. This can lead to situations where, intuitively, orthogonal variation exists but the algorithm is unable to characterize it.

**Conditions on \( \textbf{X} \) and \( \textbf{Y} \)**

We now describe the technical properties of \( \textbf{X} \) and \( \textbf{Y} \) that can lead to a situation in which O-PLS is unable to identify \( \textbf{Y} \)-orthogonal variation. A less general form of the conditions is presented first, both because it is easier to understand and because it is likely to occur in real data. For simplicity, we assume that the mean centered data is of full rank, though all of the arguments may be easily modified in the event that mean centering reduces the matrix ranks by one. Proofs are available in the Appendix.

(a) Let \( \textbf{X} \in \mathbb{R}^{n \times p} \) and \( \textbf{Y} \in \mathbb{R}^{n \times m} \) be two data matrices containing \( n \) samples of \( p \) independent and \( m \) response variables, respectively. If \( m, p \geq n \), and that \( \text{rank}(\textbf{X}) = \text{rank}(\textbf{Y}) = n \) with all \( n \) components of \( \text{PCA}(\textbf{K}) \) considered significant for O-PLS, then \( \textbf{E}_{xy} = \mathbf{0} \) and O-PLS fails to find \( \textbf{Y} \)-orthogonal variation.

A more general statement is the following.

(b) If \( \text{colspace}(\textbf{X}) \subseteq \text{colspace}(\textbf{Y}) \) and \( r_x = \text{rank}(\textbf{X}) \) components of \( \text{PCA}(\textbf{K}) \) are considered significant, then \( \textbf{E}_{xy} = \mathbf{0} \) and O-PLS fails to find \( \textbf{Y} \)-orthogonal variation.

We note that matrices satisfying the conditions in (a) automatically satisfy those in (b), so (a) is technically a corollary of (b). To see this, suppose the conditions in (a) are satisfied. Then \( \text{colspace}(\textbf{X}) = \text{colspace}(\textbf{Y}) = \mathbb{R}^n \) because both matrices are of full rank and \( r_x = n \) components of \( \text{PCA}(\textbf{K}) \) are considered significant.

An important question for the application of O-PLS is how often can one expect to encounter the above conditions in real data. In calibration studies with a relatively small number of analytes, it is unlikely that one will encounter either set of conditions. In particular, with \( m < n \leq p \), O-PLS can potentially find orthogonal variation. The conditions in (a) are impossible by default because \( \text{rank}(\textbf{Y}) < n \). And in order for the conditions in (b) to be true the spectra would have to be extremely simple, with all relevant \( \textbf{X} \)-variables being described by linear combinations of the measured concentrations. Except for the very simplest data this is unlikely to happen.
The situation is quite different, however, when the number of response variables is greater than the number of samples, which is especially true if one is trying to correlate, e.g., transcriptomic data with metabolomic data [30]. In this case, both matrices will almost certainly be of maximal rank $n$. The question then becomes whether or not the covariance matrix $K$ will also be of rank $n$. Heuristically, this can happen when $n$ or more significant factors change independently across the samples. Under these conditions, O-PLS may be unable to identify systematic variation.

In some sense, looking for orthogonal variation when the feature space is exceptionally large appears to be fundamentally inappropriate. If, instead, we start with the assumption that a relatively small number of factors are cross-correlated in an interesting way, we open the door for a host of other successful approaches. For example, one may try to enforce sparsity of the regression coefficients using a LASSO-like algorithm [132] or, as in Canonical Correlation Analysis, attempt to identify linear combinations of the features in $X$ and $Y$ simultaneously that best describe the correlations between them [72].

C.3.2 A simple example

We illustrate with an example in which there is no $Y$-orthogonal variation in $X$, yet there are aspects of $X$ that one’s intuition says should be filtered. Imagine a study in which five strains of a microorganism each has a unique genetic signature, resulting in a compound that is apparent in its Mid-IR spectrum. The aim of the study is to try to uncover information about both the genetic pathways involved and the corresponding changes in cell chemistry. Now suppose the data shows that exactly one distinct gene in the microarray is expressed, and that the expression is highly correlated with one distinct peak in the Mid-IR spectrum (C.4). If we assume mean centering of the data, then $X$, $Y$, and the covariance matrix $K$ are of maximal rank $n$. Because each gene is independent of (and hence uncorrelated with) the others, and has significant correlations with one peak in the spectrum, one cannot discard loadings of $K$ without discarding essential information about the quantities of interest. As a result, a full $n$-component model is needed.
Figure C.4: An illustration of how O-PLS ability to filter orthogonal variation depends on properties of the data matrices. **Top panel:** Data consisting of only $n = 5$ samples of gene expression $Y$ (left) is assumed to have a spurious peak on the right side of the measured spectra $X$ (center) that is truly uncorrelated with gene expression. The microarray heat maps show gene expression, increasing in value from black to white. Each gene has a distinct signature, as indicated by the corresponding letters on the $x$-axis. Solid lines show the true spectra, while dashed lines indicate systematic variation that is uncorrelated with $Y$. O-PLS does not identify orthogonal variation, and the regression coefficients $\beta$ (right) are adversely affected by the systematic variation. **Bottom panel:** If each measurement is done twice, the orthogonality of the spurious peak becomes more apparent (left). O-PLS can then identify and filter the spectral data (center) such that the orthogonal variation is removed (right).

Now suppose the samples in this experiment have been contaminated, with the result that a spurious peak appears in the spectral measurements. Although we may intuitively expect this peak to be uncorrelated with gene expression, the above statements about the rank of $X$, $Y$, and $K$ are still valid. Hence the
conditions in Theorem (a) are met and O-PLS will not filter the variation. In this case, PLS and O-PLS models are identical and contain significant contributions from the spurious peak.

One way to circumvent the inability of O-PLS to identify $\mathbf{Y}$-orthogonal variation is to increase the number of observations. Geometrically, the five genes plus systematic variation in our small example are six independent factors that the algorithm is attempting to describe by scores in a 5-dimensional subspace. Increasing the sample size, then, results not only in more accurate sample estimates to covariance (collinearity) and orthogonality (uncorrelatedness) between the variables, but also increases the dimensionality of the vector subspace used to describe true variation in the data. Indeed, if two observations of each experimental state are taken instead of one, O-PLS coefficients are seen to correspond nicely with the correct spectral peaks, and the orthogonal loading (not shown) corresponds directly to the spurious peak.

Another way to avoid this issue is to force O-PLS to identify $\mathbf{Y}$-orthogonal variation by building a separate model for each of the genes. In the example, for instance, this leads to the spurious peak being identified as orthogonal to each $\{y_i\}_{i=1}^m$. Unfortunately, one is left with $m$ separate models of the same variance in $\mathbf{X}$, as well as up to $m$ independent sets of latent variables. If the type of orthogonal variation is known a priori, one could try to use O-PLS in a somewhat surgical way via categorical response variables designed to target specific system variation [29]. This approach is not general, however, because it depends on a fairly specific knowledge of the confounding effects in the system.

C.4 Conclusion

For spectral data obeying Beer’s law in the noise free, large $n$ setting, O-PLS often generates regression models that are as good at prediction as PLS but with far fewer components. The extent to which O-PLS separates analyte and background constituents depends primarily on the covariance structure between the two groups. At its best, when the concentrations of interest are uncorrelated with the background, O-PLS loadings and regression coefficients are linear combinations of the pure profiles. Otherwise, while prediction scores remain collinear with the calibration concentrations, interpretability of loadings and regression coefficients is degraded because they contain contributions from the background profiles. The extent of the degradation depends in a complex way on concentration covariance and spectral interference of the pure
profiles.

For noise- and error-free data, PLS prediction is known to be optimal with respect to RMSEP. However, components of $X$ uncorrelated with the calibration concentrations or, more generally, those whose loadings are orthogonal to the net analyte signal vector, do not affect the optimality of PLS prediction. Because O-PLS explicitly filters scores that are $Y$-orthogonal, this suggests a reason for the similarity of prediction between the two algorithms.

Despite the success of the algorithm in the context of calibration studies, caution is needed when applying O-PLS to data sets with large feature spaces (for which both $p, m \geq n$), a situation that may be commonly encountered in the context of systems biology. In this scenario, it is likely that O-PLS will be unable to identify $Y$-orthogonal variation as defined by the algorithm. The issue may be remedied by increasing the number of samples. While doing so in chemometrics may be relatively quick and inexpensive, the same cannot be said for certain types of experiments in systems biology (e.g., microarray) in which the number of observations may be unavoidably small. If prediction is the aim, identification of $Y$-orthogonal variation may not be crucial, though the lack of theoretical models to describe the data makes evaluation of prediction quality completely dependent on data-driven methods.

If O-PLS or PLS are instead used for the exploration of features in large systems biology data sets, the algorithms are likely to lead to factorizations of the data that are difficult to interpret. Pre-processing by clustering or variable selection may improve the situation by giving the algorithms smaller, more manageable subsets on which to work. However, the search for single vectors explaining maximal variance tends to conflate intuitively independent features. Depending on the application, it may be beneficial to instead incorporate specific knowledge directly into whichever algorithm is being used, be it non-negativity of the spectral matrices [113, 92], statistical independence of the underlying factors [79, 93], or sparsity of the data [132, 75].

C.5 Appendix

Below are given the proofs of Theorems (a) and (b).

(a) Because both $X$ and $Y$ are assumed to be of rank $n$ and $m, p \geq n$, $K$ is of rank $n$ and, by
assumption that all PCA components of $K$ are significant, we can write $K = CW^T$ from Equation C.4 with $A = n$. A standard result of linear algebra is that the dimension of the row space of $X$ is equal to the matrix rank $n$, so we just need to show that each column of $W$ belongs to the row space of $X$. Then the columns of $W$ must form an orthonormal basis for the row space of $X$ and the result follows.

We show that the columns of $W$ belong to the row space of $X$ by contradiction. Assume that column $i$ of $W$ is orthogonal to the row space of $X$. Multiplying both sides of the Equation (C.4) by $XT$ from the right we see on the RHS that $XX^T$ is an $n \times n$ matrix of rank $n$, implying that $KX^T = Y^TXX^T$ is also of rank $n$. On the LHS, the $n \times n$ matrix $W^TX^T$ is of rank at most $n - 1$ because row $i$ is comprised of all zeros (by the orthogonality of row $i$ of $W$ to all rows of $X$). This implies $KX^T = CW^TX^T$ is of rank at most $n - 1$, a contradiction. Therefore, all of the columns of $W$ belong to the row space of $X$ and, because they are orthonormal and $\text{rank}(X) = n$, form an orthonormal basis for the space. Hence $XWW^T = X$, and $E_{xy} = 0$.

(b) The assumption that the column space of $X$ belongs to the column space of $Y$ implies that $r_y = \text{rank}(Y) \geq r_x$, which in turn implies that $\text{rank}(K) = r_x$. Because all $r_x$ of these components are assumed to be significant, the loading matrix $W$ contains $r_x$ orthonormal basis vectors in its columns. On account of the fact that $\text{dim(rowspace}(X)) = \text{dim(colspace}(X)) = r_x$, the columns of $W$ form an orthonormal basis for the row space of $X$. The proof of this is identical to the one above (i.e. assume at least one column of $W$ is orthogonal to the row space of $X$, and show that this leads to a contradiction about the rank of $KX^T$). It follows again that $XWW^T = X$ and $E_{xy} = 0$.

**Acknowledgement.** We acknowledge the contributions of Kofi Adragni, Jinsuk Lee, and Terry Haut for many helpful discussions, as well as Ryan Elmore for a critical review of the manuscript. This research was supported the DOE Office of Biological and Environmental Research, Grant Award No. DE-AC36-08GO28308, and by the BioEnergy Research Center. The BioEnergy Research Center is a U.S. Department of Energy Bioenergy Research Center supported by the Office of Biological and Environmental Research in the DOE Office of Science. Additional support was provided by a grant from the Department of Energy, Office of Science, SciDAC and GTL programs and by the ASCR and BioEnergy Research Offices within the
Office of Science, Grand Award No. DE-AC36-99GO10337.
Bibliography


[22] R. Bro. The N-way toolbox for MATLAB.


