Symbolic Computation of Multidimensional Fenchel Conjugates

Jonathan M. Borwein
Faculty of Computer Science
Dalhousie University
6050 University Avenue
Halifax, Nova Scotia, Canada
jborwein@cs.dal.ca

Chris H. Hamilton
Faculty of Computer Science
Dalhousie University
6050 University Avenue
Halifax, Nova Scotia, Canada
chamilton@cs.dal.ca

ABSTRACT
Convex optimization is a branch of mathematics dealing with non-linear optimization problems with additional geometric structure. This area has been the focus of considerable recent research due to the fact that convex optimization problems are scalable and can be efficiently solved by interior-point methods. Over the last ten years or so, convex optimization has found new applications in many areas including control theory, signal processing, communications and networks, circuit design, data analysis and finance.

Of key importance in convex optimization is the notion of duality, and in particular that of Fenchel duality. This work explores algorithms for calculating symbolic Fenchel conjugates of a class of real-valued functions defined on $\mathbb{R}^n$, and extends earlier work to the non-separable multi-dimensional case.

Categories and Subject Descriptors
I.1 [Symbolic and Algebraic Manipulation]: Algorithms; G.1.6 [Numerical Analysis]: Optimization—Convex Programming, Semi-definite Programming

General Terms
Convex analysis, convex functions, convex optimization

Keywords
Fenchel conjugate, Fenchel-Legendre transform, subdifferential, subgradient

1. INTRODUCTION
The area of convex optimization has been a focus of much recent research due to the wide range of problems that may be cast in the convex optimization framework, and the efficiency of algorithms for solving them. Of particular importance in the domain is the notion of Fenchel duality, as many optimization problems are easier to solve in the dual space. A necessary task in finding the dual problem is the operation of Fenchel conjugation. Previous research by Brezis [9] and Lucet [13, 14] has focussed on fast numerical methods for approximating the Fenchel conjugate. While efficient, these algorithms introduce another layer of numerical inaccuracy when solving optimization problems through dual methods.

More recent work by Bauschke and von Mohrenschildt [3, 4] developed algorithms for symbolically calculating Fenchel conjugates of one-dimensional functions. The work presented here is the result of applying several incremental improvements to the original one-dimensional algorithms, and extending the algorithm to the non-separable multi-dimensional case. This paper concentrates on the details of the underlying algorithms, while a companion paper [5] focusses on the utility of the algorithms from the point of view of a convex optimization practitioner.

To make the discussion available to the widest possible audience we begin with a primer on convex analysis. Due to space constraints we omit most proofs, instead pointing readers to the appropriate texts.

1.1 Definitions and Basic Results
Suppose $f$ is a function defined on $\mathbb{R}^n$ that takes on values in $(-\infty, \infty] = \mathbb{R} \cup \{\infty\}$. We say that $f$ is convex if

$$f(\lambda x_1 + (1 - \lambda) x_2) \leq \lambda f(x_1) + (1 - \lambda) f(x_2),$$

for every $x_1, x_2 \in \mathbb{R}^n$, and all $\lambda \in [0, 1]$. Also, we define the effective domain of $f$, dom $f$, as the set of all points where $f$ is finite-valued. A convex function is called proper if it has a non-empty domain. Convex functions lie at the heart of convex, functional and real analysis, as well convex optimization. Several excellent overviews of the subject are available, ranging from Rockafeller’s [16] and Luenberger’s [15] classics, to more modern treatments by Boyd and Vandenberghe [8] and by Borwein and Lewis [6].

Basic calculus teaches that a minimizer $\bar{x}$ of a differentiable function $f$ is necessarily a critical point: $\nabla f(\bar{x}) = 0$. Since many interesting convex functions are not everywhere differ-
entiable, this technique is not available. Instead, one defines a generalized differential, the subdifferential of $f$ at $x$ by
\[ \partial f(x) = \{ y \in \mathbb{R}^n : (y, x') - f(x') \leq f(x') - f(x), \forall x' \in \mathbb{R}^n \}. \]

We use the notation $\langle \cdot, \cdot \rangle$ to represent the standard inner product on $\mathbb{R}^n$. Also note that $\partial$ is used to represent the subdifferential, not to be confused with a partial derivative. Members of this subdifferential are called subgradients, and have a clear geometric interpretation as being slopes of tangential hyperplanes that minorize $f$ at the point $x$. The importance of subgradients in convex optimization stems from the calculus-like fact that $\bar{x}$ is a global minimizer of $f$ if and only if $0 \in \partial f(\bar{x})$. We state some important basic results, the proofs of which may be found in any of [11], [15] or [16].

**Theorem 1.1 (Subdifferentials and Derivatives).** Consider a convex function $f : \mathbb{R}^n \to \mathbb{R}$. The subdifferential generalizes differentiability: $f$ is differentiable at $x$ if and only if $\partial f(x)$ is a singleton, in which case $\partial f(x) = \{ \nabla f(x) \}$.\[ \]

**Theorem 1.2 (Subdifferential on $\mathbb{R}$).** If $f$ is convex and defined on $\mathbb{R}^n$, then the left ($f^\prime_-$) and right ($f^\prime_+$) derivatives exist at every point in dom $f$. Moreover, the subdifferential at every $x \in$ dom $f$ is a closed interval or singleton completely described by the directional derivatives:
\[ \partial f(x) = [f^\prime_-, f^\prime_+]. \]

The Fenchel conjugate, or Fenchel-Legendre transform, of $f$, denoted $f^\star$, is defined by:
\[ f^\star (y) = \sup_{x \in \mathbb{R}^n} \{ \langle y, x \rangle - f(x) \}, \quad \forall y \in \mathbb{R}^n. \]

The Fenchel conjugate is always a convex and lower semi-continuous (but not necessarily proper, hence possibly everywhere $\pm \infty$) function on $\mathbb{R}^n$. A function $f$ is called lower semi-continuous, or lsc, at a point $x$ if
\[ \liminf_{y \to x} f(y) \geq f(x). \]

The role of the Fenchel conjugate in convex analysis is extremely important, and draws many parallels with the role of the Fourier transform in harmonic analysis. Assuming lower semi-continuity and properness of $f$, the bi-conjugate of $f$ recovers the original function: $f = f^{\star \star}$. In fact, the converse is also true, leading to:
\[ f \text{ is convex, proper and lsc } \iff f = f^{\star \star}. \]

An immediate consequence of the definition of the Fenchel conjugate is the well-known Fenchel-Young Inequality:
\[ f(x) + f^\star(y) \geq \langle x, y \rangle, \quad \forall x, y \in \mathbb{R}^n. \tag{1} \]

**1.2 Optimization and Duality Results**

We present a few duality results highlighting the natural importance of Fenchel conjugates.

### 1.2.1 Convex Optimization Duality

Convex optimization deals with primal problems of the type
\[ p = \inf_{x \in \mathbb{R}^n} \{ f(x) + g(Ax) \}, \]

where $A \in \mathbb{R}^{m \times n}$, $f$ is convex and lsc on $\mathbb{R}^n$, and $g$ is convex and lsc on $\mathbb{R}^m$. Fenchel conjugation leads to a natural dual representation of the problem as
\[ d = \sup_{z \in \mathbb{R}^m} \{ f^\star(-A^\top z) + g^\star(z) \}, \]

where $A^\top$ is simply the transpose of $A$. There are several fundamental results relating these dual formulations, a few of which are listed below.

**Theorem 1.3 (Fenchel’s Duality Theorem).** Suppose $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$ and $f$, $g$ and $A$ are as above. Then the following hold:

1. **Weak Duality:** $p \geq d$.
2. **Strong Duality:** If $A(\text{dom } f) \cap \text{int dom } g \neq \emptyset$, then $p = d$ and the infimum defining $d$ is attained.
3. **Primal Solutions:** If $z$ is a solution to the dual, then the solutions to the primal are equal to the (possibly empty) set
\[ A^{-1} \partial g^\star(z) \cap \partial f^\star(-A^\top z). \]

**Proof.** Refer to [6]. \(\square\)

Fenchel’s Duality Theorem highlights the importance of subdifferentials and Fenchel conjugates. Additionally, it is worth noting that Fenchel Duality is general enough to encompass both Linear Programming Duality and the well known Min-Max Theorem from game theory. For further details refer to [15].

### 1.2.2 Semi-Definite Optimization

In semi-definite optimization one minimizes a linear function subject to the constraint that an affine combination of symmetric matrices remains positive semi-definite. Such a constraint is inherently non-linear and non-smooth, but convex. Semi-definite optimization is a generalization and a unification of several standard problems (linear, quadratic and convex optimization) and as such finds many applications. Although much more general than linear programming, they are not much harder to solve.

In semi-definite optimization the primal problem may be stated as
\[ \inf_{x \in \mathbb{R}^n} \{ c^T x : B(x) \preceq 0 \}, \]

where $B(x) = B_0 + \sum_{i=1}^{m} x_i B_i$, $c \in \mathbb{R}^m$ and $B_i \in \mathbb{R}^{n \times n}$. General duality results for semi-definite programs are weaker than for linear programs and no straight-forward simplex methods exist for solving them. Generally, they are solved using path-following interior point methods, which instead optimize a convergent sequence of smooth approximations.
to the original problem. Under this framework, a smoothing convex barrier function $H$ is introduced leading to the approximate primal problem

$$\inf_{x \in \mathbb{R}^n} \{ c^T x + r^{-1} H(rB(x)) \}$$

which converges to the original as $r \to \infty$. The dual of the approximate problem is given as

$$\sup_{Z \succcurlyeq 0} \{ \text{tr}(B_0Z) - rH^*(Z) : -\text{tr}(B_iZ) = c_i \}.$$ 

As can be seen, the Fenchel conjugate plays an important role in the dual formulation. For much more detail, refer to [2, 6, 8].

2. ALGORITHMIC APPROACH

2.1 One Dimension

Most of the ideas presented in this section closely follow work by Heinz Bauschke and Martin von Mohrenschildt in [3, 4].

2.1.1 A Good Class of Functions

Computer algebra systems are naturally suited to working with functions defined over the real numbers that are finite in representation. It is useful to characterize what is meant by having a finite representation, and to formalize the space of admissible functions.

Let $F$ be the class of all functions $f$ satisfying the following conditions:

(i) $f$ is a function from $\mathbb{R}$ to $\mathbb{R}$;
(ii) $f$ is a closed convex function;
(iii) $f$ is continuous on its effective domain; and,
(iv) there are finitely many points $x_i$ such that

$$x_0 = -\infty < x_1 < \cdots < x_n < x_{n+1} = \infty$$

and each $f|_{(x_i, x_{i+1})}$ is one of the following:

(a) identically equal to $\infty$; or,

(b) differentiable.

Since we restrict ourselves to functions on the real line, condition (iii) is equivalent to the lower semi-continuity of $f$. Additionally, if $f_1, f_2 \in F$ and $\alpha_1, \alpha_2 \geq 0$, then $\alpha_1 f_1 + \alpha_2 f_2$ and $f^*$ are in $F$. The class $F$ is thus well-suited to our purpose.

2.1.2 Subdifferentiation

A convex lower semi-continuous function function on the real line is very well behaved; for instance, it is subdifferentiable on the interior of its domain. It is a straight-forward application of Theorems 1.1 and 1.2 to calculate the subdifferential. Over each open-interval the subdifferential is calculated as

$$\partial f|_i = \begin{cases} \emptyset, & \text{if } f|_i = \infty, \\ \{ f|_i \}, & \text{otherwise.} \end{cases}$$

For each point $x_i \notin \text{dom } f$ the subdifferential is empty, while for each point $x_i \in \text{int } \text{dom } f$ it may be calculated as the (possibly singleton) closed interval

$$\partial f(x_i) = \lim_{x \downarrow x_i} f|_{x_i - 1}(x), \lim_{x \uparrow x_i} f|_{x_i + 1}(x).$$

We are left with calculating the subdifferential over the (zero, one or two) points $x_i \in \text{bd } \text{dom } f$ as

$$\partial f(x_i) = \begin{cases} (-\infty, \infty), & \text{if } f|_{x_i - 1} = \infty = f|_i; \\
\infty, \lim_{x \downarrow x_i} f|_{x_i}(x), & \text{if } f|_{x_i - 1} = \infty \neq f|_i; \text{ or,}
\lim_{x \uparrow x_i} f|_{x_i - 1}(x), \infty, & \text{if } f|_{x_i - 1} \neq \infty = f|_i. \end{cases}$$

2.1.3 Fenchel Conjugation

Given the subdifferentiable of $f$ the conjugate of a point $y$ may be calculated in two steps: firstly, solve $y \not\in \partial f(x)$ for $x$ – the key step – and let $\bar{x}$ be such a solution (if none can be found then $y \not\in \text{dom } f^*$, and $f^*(y) = \infty$. Secondly, use the Fenchel-Young equality (Equation 1) to obtain

$$f^*(y) = \langle \bar{x}, y \rangle - f(\bar{x}).$$

To calculate the conjugate over the whole real line, we simply invert each $f|_i$ and insert it into the Fenchel-Young inequality, which in turn defines the conjugate for all $y \in (\lim_{x \downarrow x_i} f|_{x_i}(x), \lim_{x \uparrow x_i} f|_{x_i + 1}(x))$. Similarly, each point $x_i$ with a non-singleton differential $\partial f(x_i) = (a, b)$ yields $f^*(y) = x_i y - f(x_i)$ for $y \in (a, b)$. Continuity then determines the value of the Fenchel conjugate at boundary points, with the function taking the value $\infty$ outside of its domain.

2.1.4 Inverting the Subdifferential

Inverting the component derivatives is the biggest challenge to symbolically computing the Fenchel conjugate. We rely on the Maple function solve, which by its nature has to deal with branch cuts and hence does not always return a unique closed form inverse.

A typical example of this behaviour is in conjugating $f = x^3/4$. In order to proceed we need to invert $y = f'(x) = x^3$. Since Maple implicitly works in the complex plane an inverse calculation yields the three cubic roots of $y$. But none of these expressions is the real root for all real values of $y$.

**Definition 2.1** (Branch point). For our purposes, a branch point is a point at which a branch cut of an analytic multi-valued function intersects the real line.

If Maple had internal representations of elementary functions (and their inverses) as functions from $\mathbb{R}$ to $\mathbb{R}$, this problem would largely disappear. Prior to Maple 10, this functionality did not exist in any form. Maple 10 includes a limited ability to work purely in the real domain (the RealDomain package), although it is somewhat restricted and not yet mature enough for our needs. As these underlying tools mature, so will our algorithms.

Thus, we are left with having to explicitly find the branch points and determine which branch of the inverse is applicable over which sub-interval. Conveniently, classical complex analysis tells us exactly where these branch points may be located when the function we are inverting is analytic.
**Theorem 2.2** (Existence of Complex Roots).
Suppose that \( f(z) \) is analytic at \( z_0, f(z_0) = w_0 \), and that \( f(z) - w_0 \) has a zero of order \( n \) at \( z_0 \). If \( \epsilon > 0 \) is sufficiently small, there exists a corresponding \( \delta > 0 \) such that for all \( z \) with \( |z - z_0| < \delta \) the equation \( f(z) = w \) has exactly \( n \) roots in the disk \( |z - z_0| < \epsilon \).

**Proof.** We refer the reader to Chapter 3, Theorem 11 of [1]. \( \square \)

**Corollary 2.3** (Location of Branch Points).
Suppose that \( f(z) \) is analytic on the entire neighborhood \( |z - z_0| < \epsilon \), and let \( g_1(a), \ldots, g_n(a) \) represent the \( n \) roots of \( f(z) = a \) on the neighborhood \( |a - w_0| < \delta \). Then \( g_1(w_0) = \cdots = g_n(w_0) = z_0 \).

**Proof.** Due to the \( n \)th order zero of \( f(z) \) at \( z_0 \), it follows that \( f(z) \) may be expressed as \( f(z) = (z - z_0)^n g(z) \), where \( g(z) \neq 0 \), for all \( z \) with \( |z - z_0| < \epsilon \). Due to the analyticity of \( f(z) \) and the existence of exactly \( n \) roots by Theorem 2.2, for any \( a \) with \( |a - w_0| < \delta \) we can write \( f(z) - \alpha = (z - g_1(a)) \cdots (z - g_n(a)) h(z) \), for some \( h(z) \neq 0 \). Since \( \lim_{a \to w_0} h(z) = (z - z_0)^n g(z) \), and therefore

\[
[\lim_{a \to w_0} (z - g_1(a)) \cdots (z - g_n(a)) h(z) = (z - z_0)^n g(z),
\]

we introduce the concept of a partial conjugate. Consider an \( n \)-dimensional function that has had a one-dimensional conjugate calculated with respect to the variable \( x_1 \). The notation \( f^{x_1} \) then represents this partial conjugate of \( f \) with respect to \( x_1 \). The above may be rewritten as

\[
f^* = \left(-\left((f^{x_1})^{x_2}\right)^x_3\right)^x_1.
\]

This implies that an \( n \)-dimensional conjugate can be seen as an iterated sequence of \( n \) one-dimensional conjugation and \( n - 1 \) negation operations.

### 2.2 Many Dimensions

For functions defined over \( \mathbb{R}^n \) the Fenchel conjugate may be rewritten as:

\[
f^*(y) = \sup_{x} \{ (x, y) - f(x) \}
\]

\[
= \sup_{x_1, \ldots, x_n} \left\{ \sum_{i=1}^{n} x_i y_i - f(x) \right\}
\]

\[
= \sup_{x_1} x_1 y_1 + \cdots + \sup_{x_n} \{ x_n y_n - f(x) \} \cdots .
\]

As an immediate result of Corollary 2.3 we can infer that branch points of a function \( f \) may only occur at zeroes of the first derivative of \( f \). Assuming we are able to find these zeroes, and there are finitely many of them within our interval of interest, we may then test each of the candidate inverses over each implied sub-interval and determine which is the correct branch. Finding these zeroes is left to Maple's routine `solve`, which may not always succeed.

The original algorithms presented in [3, 4] have no facilities to deal with branch selection. The above extension to these original algorithms greatly enlarges the space of functions over which the algorithm may successfully calculate conjugates.

### 2.2.1 A Good Class of Functions

The natural space to work in is the recursive extension to \( F \). An \( n \)-dimensional function \( f \) is in \( F^n \) if:

(i) \( f(x_1, \ldots, x_n) \) is a function from \( \mathbb{R}^n \) to \( \mathbb{R} \);
(ii) \( f(x_1, \ldots, x_n) \) is a closed convex function;
(iii) \( f(x_1, \ldots, x_n) \) is continuous on its effective domain; and,
(iv) there are finitely many points \( a_i \) such that \( a_0 = -\infty < a_1 < \cdots < a_{n-1} < a_n = \infty \) and \( f \) restricted to each open interval \( (a_i, a_{i+1}) \) is in \( F^{n-1} \) (where \( F^1 = F \)) with respect to the variables \( x_2, \ldots, x_n \).

### 2.2.2 Fenchel Conjugation

Functions in \( F^n \) have an implicit variable order due to their structure. A function defined over the variable order \( x_1, \ldots, x_n \) may only be partially conjugated along the last variable, \( x_n \). In order to calculate a partial conjugate with respect to another variable \( x_j \), the function must be rewritten such that \( x_j \) is the last variable in its representation. We illustrate with an example in \( F^2 \).

**Example 2.4** (Product of roots). Consider the two-dimensional function

\[
f(x_1, x_2) = \begin{cases} 
\infty, & x_2 \leq x_1 < 0 \\
\infty, & x_2 < 0 \\
0, & x_2 = 0, \quad x_1 = 0 \\
0, & 0 < x_2 < x_1 \\
\infty, & x_2 = 0, \quad 0 < x_1 \end{cases}
\]

Calculating the partial conjugate with respect to the \( x_2 \) axis involves calculating two one-dimensional partial conjugates; one along the line \( x_1 = 0 \) and the other over the half-plane \( 0 < x_1 \). Calculating these conjugates (and negating the results) yields:

\[-f^{x_2}(y_2, x_1) = \begin{cases} 
\infty, & y_2 \leq 0, \quad x_1 < 0 \\
\infty, & y_2 < 0, \quad x_1 = 0 \\
\infty, & y_2 = 0, \quad 0 < x_1 \end{cases}
\]

We now wish to calculate the partial conjugate along the \( x_1 \) variable in order to complete the two-dimensional conjugation. However, in order to do this, we must first reorder the variables to \( (y_2, x_1) \). In this example this is easily done.
through inspection, resulting in:

\[-f'^2(y_2,x_1) = \begin{cases} 
\infty, & x_1 < 0 \\
0, & x_1 = 0, \ y_2 < 0 \\
\frac{y_1}{y_2}, & 0 < x_1 \\
\infty, & \forall x_1, \ y_2 = 0 \\
\infty, & \forall x_1, \ y_2 < 0
\end{cases}\]

We may now proceed to calculate the complete conjugate by partially conjugating along the x₁ axis. There are two distinct one-dimensional conjugates to be calculated along the line y₂ = 0 and the half-plane y₂ < 0. This yields:

\[f^*(y_2, y_1) = \begin{cases} 
0, & y_1 < \frac{1}{y_2} \\
0, & y_1 = \frac{1}{y_2}, \ y_2 < 0 \\
\infty, & \frac{1}{y_2} < y_1 \\
\infty, & \forall y_1, \ y_2 = 0 \\
\infty, & \forall y_1, \ y_2 < 0
\end{cases}\]

It is often desirable to have the conjugated function in the same variable order as the original function. This involves yet another variable reordering to \((y_1, y_2)\). The result of this operation is the final conjugate:

\[f^*(y_1, y_2) = \begin{cases} 
0, & y_2 < \frac{1}{y_1} \\
0, & y_2 = \frac{1}{y_1}, \ y_1 < 0 \\
\infty, & \frac{1}{y_1} < y_2 \\
\infty, & \forall y_2, \ y_1 = 0 \\
\infty, & \forall y_2, \ 0 < y_1
\end{cases}\]

2.2.3 Variable Reordering

A function \(f \in F^n\) can be thought of as a union of functions \(f_i\) defined over disjoint sets \(S_i \subset \mathbb{R}^n\). Given the variable order \(x_1, \ldots, x_n\), each \(S_i\) is naturally represented as \(S_i = \{x : x_1 \in X_1, x_2 \in X_2(x_1), \ldots, x_n \in X_n(x_1, \ldots, x_{n-1})\}\). The operation of changing variable orders to (for example) \(x_n, x_1, \ldots, x_{n-1}\) is equivalent to finding \(X_i\) such that \(S_i = \{x : x_n \in X_n, x_1 \in X_1(x_n), \ldots, x_{n-1} \in X_{n-1}(x_n, x_1, \ldots, x_{n-2})\}\). This is completely equivalent to the problem of changing the order of variables in a multiple-integral (assuming the integrand is sufficiently well behaved). Consider the integral

\[
\int_{S_i} f(x) dx = \int_{X_1} \cdots \int_{X_n} f(x) dx_{n-1} \cdots dx_1.
\]

To change the order of the variables to \(x_n, x_1, \ldots, x_{n-1}\) we wish to find \(X_i\) such that:

\[
\int_{S_i} f(x) dx = \int_{X_n} \int_{X_{n-1}} \cdots \int_{X_1} f(x) dx_{n-1} \cdots dx_1 dx_n.
\]

In the general case (where none of the dimensions describing \(S_i\) are separable) this problem is extremely hard. However, in the non-separable two-dimensional case the problem may be fully broken down to a set of 23 distinct sub-problems (which may be further reduced to 12 due to symmetry), each of which may be solved assuming appropriate zeroes and inverses may be found ([11]). This allows the multi-dimensional symbolic Fenchel conjugation algorithm to deal quite robustly with non-separable two-dimensional objects, and higher dimensional ones in certain cases.

3. EXAMPLES AND APPLICATIONS

3.1 The Maple Package SCAT

Earlier work by Bauschke and Mohrnschildt [3, 4] focussed on symbolically calculating exact subdifferentials and conjugates for one-dimensional real-valued functions on \(\mathbb{R}\), and separable multi-dimensional function on \(\mathbb{R}^n\). Their work led to the development of the Maple package fenchel. The Maple package SCAT is the result of refining that work and extending it to the non-separable many-dimensional case. It also serves to unite the complementary approach of numerically computing subdifferentials and conjugates, using approaches such as those developed in [9, 13, 14] when symbolic approaches break down.

The Maple package SCAT (Symbolic Convex Analysis Toolkit) introduces several new constructs and commands to Maple: the objects PWF and SD for representing convex functions and subdifferentials; the function SCAT[Plot] for exploring them graphically; the function SCAT[Eval] for evaluating them at points, or taking lower dimensional slices; the functions SCAT[SubDiff] and SCAT[Int] for calculating subdifferentials from convex functions and vice-versa; and the functions SCAT[Conj] and SCAT[IntConv] for calculating Fenchel conjugates and infimal convolutions. Additionally, the toolkit is well integrated with Maple, tying in with Maple’s conversion (convert), evaluation (eval, evalf, etc), pretty printing (print) and simplification (simplify) functionality.

The latest version of this software, along with extensive documentation and usage guides, are available at:

http://drive.cs.dal.ca/projects/scat/

3.2 Classic Examples

We explore the functionality and capabilities of SCAT using several classic examples from the literature.

Example 3.1 (Absolute value). One of the simplest examples of a convex function that is not everywhere differentiable is the absolute value function \(f : x \mapsto |x|\). Its derivative at the origin fails to exist since \(f'(0) = -1 < 1 = f'(0)\). The notion of the subgradient is able to capture this behaviour and accordingly it is seen that \(\partial f(0) = [-1, 1]\). In order to explore this function we first represent it in a form that SCAT understands; the PWF (piecewise function) format:

\[
> f1 := \text{convert}(|x|, \text{PWF});
\]

\[
f1 := \begin{cases} 
-x, & x < 0 \\
0, & x = 0 \\
x, & x > 0
\end{cases}
\]

We may easily calculate the subdifferential of \(f1\):

\[
> \text{sd}f1 := \text{SubDiff}(f1);
\]

\[
\text{sd}f1 := \begin{cases} 
\{-1\}, & x < 0 \\
[-1, 1], & x = 0 \\
\{1\}, & x > 0
\end{cases}
\]

We may also calculate the conjugate, yielding:

\[
> g1 := \text{Conj}(f1, y);
\]

\[
g1 := \begin{cases} 
\infty, & y < -1 \\
0, & y = -1 \\
0, & (-1 < y) \text{ and } (y < 1) \\
0, & y = 1 \\
\infty, & 1 < y
\end{cases}
\]
Scaling and manually verify the convexity of
piecewise
the following commands:

```
> g5 := Conj(f5,y);
```

| $\begin{cases} 
-3y + 1, & y < -\frac{1}{3} \\
\frac{5}{2}, & y = -\frac{1}{3} \\
\frac{y^2 + 2y + 2}{1 + y}, & (y < \frac{1}{3}) \text{ and } (y < 0) \\
2, & y = 0 \\
0, & (0 < y) \text{ and } (y < 2) \\
2, & y = 2 \\
\frac{y^2 - 2y + 2}{1 + y}, & 2 < y 
\end{cases}$
```

| $f5$ := Subs(p=1/(1-1/q),g5);

```
> F5 := Conj(g5,x):
```

Equal(f5,F5);

```
true
```

Example 3.3 (Young’s Inequality). Suppose $1 < p < \infty$ and let $q$ be such that $\frac{1}{p} + \frac{1}{q} = 1$. The equality

$$\frac{1}{p}a^p + \frac{1}{q}b^q \geq ab, \quad \forall a, b > 0,$$

is known as Young’s Inequality. As we are about to see, since $(\frac{1}{p})^* = \frac{1}{q}$, this is actually a special case of the more general Fenchel-Young inequality from Equation 1. In this example we show and confirm the above conjugate pair. As a corollary to finding the conjugate pair, we in fact prove Young’s inequality.

This example elaborates on a similar example provided in [4]. The algorithms developed in this work are able to handle $p$ as a free parameter while those in [4] were restricted to fixed values of $p$. The general pair of conjugate functions is easily derived using the following commands:

```
> f7 := convert((abs(x)*p/p,PWF,x),{p>1});
g7 := Conj(f7,y);
g7 := Subs(p=1/(1-1/q),g7);
```

In creating $f7$, notice that we passed additional parameters consisting of a set of assumptions. In this example, if we do not provide the information that $p > 1$ then the process will fail, producing the following output:

```
> f := convert( abs(x)*p/p, PWF, x );
Error, (in EvalRel) unable to evaluate relation: 1/p*limit(x^p,x = 0,right) = 1/p*limit((-x)^p,x = 0,left)
```

### 3.3 NMR Imaging

The method of maximum entropy reconstruction has been applied in nuclear magnetic resonance spectroscopy (NMR) to the problem of estimating complex spectra. It has proven to be a useful approach as it has the ability to estimate high-resolution spectra from short data records, to deconvolve spectra without enhancing noise and to estimate spectra from non-uniformly sampled time series [7, 12]. By maximizing entropy a spectral estimate is found with the least amount of ‘false information’, thus, such an optimized spectra is the ‘most informative’ of all possible estimates.

The choice of the most appropriate information entropy to
use has been considered [12]. The Hoch and Stern information measure for complex spectra has been shown to be consistent with the underlying statistical mechanical entropy governing the physical system. In addition to this, it was shown by Borwein et al in [7] that this entropy has a very natural dual representation, and the corresponding dual optimization problem is more efficient to solve with greater numerical accuracy.

Let $D = [-\Delta/2, \Delta/2]$ be the support of the unknown spectrum, modelled by a function $\phi(s) \in L^2(D)$, and let $\psi(t) \in L^2(D)$ represent the time signal. The problem is that of recovering $\phi$ from a noisy discrete measurement of $\psi$, $Y = \psi(\delta t) \in \mathbb{C}^m$, where $\mathbb{J} = \{0, \ldots, m-1\}$ and $\delta t$ represents the sampling interval. The Hoch and Stern entropy is defined by $H(X) = \sum_{j=1}^{|\mathbb{J}|} h(X_j)$, where $h$ is the convex function on $\mathbb{C}$ given by

$$h(z) = f \left( \frac{|z|}{b} \right) \quad \text{and} \quad f(u) = u \log u + \sqrt{1 + u^2} - \sqrt{1 + u^2},$$

with $f(|z|)$ plotted in Figure 3. With these definitions one version of the primal problem may be stated as

$$\inf \{ H(X) : X \in \mathbb{C}^m, \|Y - AX\| \leq \epsilon \},$$

with the matrix $A$ representing the discrete Fourier transform operator. In [7] the dual of this problem is derived as

$$\sup \{ \Re \{ Y\Lambda \} - \epsilon \| \Lambda \| - H^*(A^*\Lambda) : \Lambda \in \mathbb{C}^m \}.$$ 

The motivation to pursue the dual problem came from the casual observation that $f^*(u^*) = \cosh u^*$ (plotted in Figure 4), which was calculated by Borwein using SCAT’s direct ancestor, fenchel. This leads immediately to $H^*(X^*) = \sum_{j=1}^{n_j} \cosh(\bar{b}|X_j^*|)$. Given a solution $\Lambda$ to the dual problem, a solution to the primal is found as

$$\bar{X} = b \exp i \text{arg} A^*\Lambda \sinh(\bar{b}|A^*\Lambda|),$$

and a solution to the original spectra as

$$\bar{\phi}(s) = b \exp i \text{arg} [A^*\Lambda](s) \sinh(b[|A^*\Lambda|(s))].$$

As can be seen the dual problem is an even hence smooth unconstrained maximization. Furthermore, the characterization of the dual of the Hoch and Stern entropy showed that it is directly related to the dual of the classical Shannon entropy, being the even part. These insights have led to the development of more efficient and numerically stable algorithms for reconstructing NMR images, and were partially facilitated by the ease and low-cost of thought experiments using tools like SCAT.

4. LIMITATIONS AND FUTURE WORK

4.1 Algorithmic Improvements

In one dimension, the biggest challenge to symbolically computing Fenchel conjugates is in inverting the subdifferential. Although improved compared to earlier versions of the algorithm, we rely on the Maple function solve to identify branch points and applicable inverses for each branch. This often prevents us from computing a closed form even when such a solution exists. As Maple (or Mathematica, in which one could certainly also implement these ideas) makes improvements in the underlying machinery, the space of functions on which SCAT can successfully compute conjugates will only grow.

In two-dimensions the same difficulty in computing inverses often prevents us from changing the variable order of the partial conjugate, preventing the completion of the conjugate operation. In higher dimensions this problem is even harder, and we currently have no mechanism to even approach it.

As mentioned in Section 2.1.4, Maple 10 includes a limited ability to perform calculations directly in the real domain, avoiding the need to deal with complex-valued functions and boundary points. Although the functionality is not mature enough to meet our current needs, it can be integrated into the existing algorithms to improve performance.

4.2 Interface Improvements

One drawback to the existing tools is that we are limited by the internal representation of functions. Functions must be input in rectangular coordinates, making it very awkward to manipulate some otherwise very simple and natural functions (for example, the indicator function of the Euclidean unit ball in $\mathbb{R}^n$). It would be interesting to explore alternative methods of function representation, including the use of different coordinate systems.

4.3 Automated Inequality Proving

Example 3.3 demonstrated that is possible to derive and prove entire class of Fenchel-Young like inequalities. In fact, there are a host of inequalities that may be derived using convex analysis. Since $f^{**} = f$ implies that $f$ is convex, we may automatically determine convexity through the calculation of a bi-conjugate. With a certificate of convexity and a conjugate in hand, other techniques becomes immediately accessible. Thus, the algorithms developed in this paper may form the base of a system that is able to de-
termine the validity of certain families of inequalities, much like the algorithms presented in [10].

5. CONCLUDING REMARKS
In the Maple package SCAT we have implemented algorithms for conjugation, subdifferentiation and infimal convolution of convex functions defined on \( \mathbb{R} \). We have extended previous algorithms to allow conjugation of non-separable functions defined on \( \mathbb{R}^2 \), and in certain cases for higher dimensions. We have provided examples and applications and commented on the limitations of the algorithms. One especially useful application is that practitioners can produce automated code for symbolic components of larger computational projects. It is our hope that SCAT will be useful to researchers, instructors and students of convex analysis and optimization.

6. REFERENCES