

Symbolic Fenchel Conjugation

Jonathan M. Borwein · Chris H. Hamilton

Received: 30 September 2005 / Accepted: 3 April 2006 / Published online: 28 April 2007
© Springer-Verlag 2007

Abstract Of key importance in convex analysis and optimization is the notion of duality, and in particular that of Fenchel duality. This work explores improvements to existing algorithms for the symbolic calculation of subdifferentials and Fenchel conjugates of convex functions defined on the real line. More importantly, these algorithms are extended to enable the symbolic calculation of Fenchel conjugates on a class of real-valued functions defined on \mathbb{R}^n . These algorithms are realized in the form of the *Maple* package *SCAT*.

Keywords Fenchel conjugate · Legendre–Fenchel transform · Subdifferential · Subgradient

Mathematics Subject Classification (2000) 44A15 · 65K99 · 68W30

1 Background and motivation

To make the development available to a wide variety of practitioners we include the following brief discussion on the basics of convex analysis.

In honour of Alfred Auslender.

J. M. Borwein (✉) · C. H. Hamilton
Faculty of Computer Science, Dalhousie University, 6050 University Avenue,
Halifax, NS, Canada, B3H 1W5
e-mail: jborwein@cs.dal.ca

C. H. Hamilton
e-mail: chamilton@cs.dal.ca

1.1 Definitions and basic results

Suppose f is a function defined on \mathbb{R}^n that takes on values in $(-\infty, \infty] = \mathbb{R} \cup \{\infty\} = \overline{\mathbb{R}}$. Recall 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 $\lambda \in [0, 1]$. Recall also that the *effective domain* of f , $\text{dom } f$, is the set of all points where f is finite-valued. 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 Rockafellar's [13] and Luenberger's [12] classics, to more modern treatments by Boyd and Vandenberghe [7] and by Borwein and Lewis [5].

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 functions are not everywhere differentiable, this leads naturally to a generalized derivative, the *subdifferential* of f at x by

$$\partial f(x) = \{y \in \mathbb{R}^n : \langle y, x' - x \rangle \leq f(x') - f(x), \forall x' \in \mathbb{R}^n\}.$$

Members of this subdifferential are called *subgradients* and are important in convex optimization due to the calculus-like fact that \bar{x} is a global minimizer of f if and only if $0 \in \partial f(\bar{x})$. The proofs of the following basic results on subdifferentials have been omitted for brevity, but may be found in any of [12, 13] or [8].

Theorem 1 (Differentiability and the Subdifferential) *Consider a convex function $f : \mathbb{R}^n \mapsto \overline{\mathbb{R}}$. Then 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 2 (Subdifferential on \mathbb{R}) *If $f : \mathbb{R} \rightarrow \overline{\mathbb{R}}$ is convex, then the left (f'_-) and right (f'_+) derivatives exist at every point in $\text{dom } f$. Moreover, for every $x \in \text{dom } f$, it follows that*

$$\partial f(x) = \{y : f'_-(x) \leq y \leq f'_+(x)\}.$$

The *Fenchel conjugate*, or *Legendre–Fenchel transform*, of f , denoted f^* , is defined by

$$f^*(y) = \sup_{x \in \mathbb{R}^n} \{\langle y, x \rangle - f(x)\}, \quad \forall y \in \mathbb{R}^n.$$

The Fenchel conjugate is always a convex and lower semi-continuous function on \mathbb{R}^n .

Assuming f is itself lower semi-continuous and proper, the bi-conjugate of f recovers the original function: $f = f^{**}$. In fact, the converse is also true, leading to:

$$f \text{ is convex and lower semi-continuous} \Leftrightarrow f = f^{**}.$$

An immediate consequence of the definition of the Fenchel conjugate is the well-known *Fenchel–Young Inequality*:

$$f(x) + f^*(y) \geq \langle x, y \rangle, \quad \forall x, y \in \mathbb{R}^n. \tag{1}$$

Sufficient and necessary conditions for this to hold with equality may be formulated in terms of subgradients:

$$f(x) + f^*(y) = \langle x, y \rangle \Leftrightarrow y \in \partial f(x) \Leftrightarrow x \in \partial f^*(y).$$

1.2 Optimization and duality results

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 lower semi-continuous on \mathbb{R}^n , and g is convex and lower semi-continuous on \mathbb{R}^m . Fenchel conjugation leads to a natural dual representation of the problem as

$$d = \sup_{z \in \mathbb{R}^m} \{f^*(-A^*z) + g^*(z)\},$$

where A^* is simply the transpose of A . There are several fundamental results relating these dual formulations, a few of which we will present here.

Theorem 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 supremum 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*

$$A^{-1} \partial g^*(z) \cap \partial f^*(-A^*z).$$

Proof Refer to [5].

Fenchel’s duality Theorem highlights the importance of subdifferentials and Fenchel conjugates. For further details refer to [12].

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.

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 $B_i \in \mathbb{R}^{n \times n}$. Generally, semi-definite programs are solved using path-following interior point methods, which 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

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

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

$$\sup_{Z \succeq 0} \{\text{tr}(B_0 Z) - rH^*(Z) : -\text{tr}(B_i Z) = c_i\}.$$

As can be seen, the Fenchel conjugate plays an important role in the dual formulation. For much more detail, refer to [2, 5, 7].

2 Algorithmic approach

In this section, we develop and demonstrate the underlying algorithms used in calculating symbolic conjugates. Section 2.1 builds upon earlier work [3, 4] and explores symbolic conjugation in one-dimension, while Sect. 2.2 presents new work [8] extending the one-dimensional algorithms to the non-separable multi-dimensional case.

2.1 One dimension

Most of the ideas presented in Sects. 2.1.1–2.1.3 closely follow those illustrated by Heinz Bauschke and Martin von Mohrenschildt in [3, 4], themselves based on earlier unpublished work by the first author. However, significant effort was placed into expanding the scope of functions which could be successfully handled, and Sect. 2.1.4 focusses on improvements to the one-dimensional algorithms.

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 \mathcal{F} be the class of all functions f satisfying the following conditions:

- (i) f is a function from \mathbb{R} to $\overline{\mathbb{R}}$;
- (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

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

and each $f|_i$ (f restricted to the open interval (x_i, x_{i+1})) is one of the following:

- (a) identically equal to ∞ ; or,
- (b) differentiable.

Since we restrict ourselves to functions on the real line, condition (iii) is equivalent to the lower semi-continuity of f . Perhaps most important is the observation that \mathcal{F} is closed under conjugation, as shown in Proposition 1. Additionally, it is straight-forward to see that \mathcal{F} is closed under non-negative scalar multiplication and addition, thus the class \mathcal{F} is particularly well-suited to our purpose.

2.1.2 Subdifferentiation

A convex lower semi-continuous 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 and 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 $x_i \in \text{int dom } f$ it may be calculated as the (possibly singleton) closed interval

$$\partial f(x_i) = \left[\lim_{x \uparrow x_i} f|_{i-1}'(x), \lim_{x \downarrow x_i} f|_i'(x) \right].$$

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

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

2.1.3 Fenchel conjugation

Given the subdifferential of f the conjugate of a point y may be calculated in two steps: firstly, solve $y \in \partial f(x)$ for x – the key step – and let \bar{x} be such a solution (if none

can be found then $y \notin \text{dom } f^*$, and $f^*(y) = \infty$). Secondly, use the Fenchel–Young equality (Eq. 1) to obtain $f^*(y) = \langle \bar{x}, y \rangle - f(\bar{x})$.

In other words, 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 $y \in (\lim_{x \downarrow x_i} f|_i'(x), \lim_{x \uparrow x_{i+1}} f|_i'(x))$. 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 ∞ outside of its effective domain.

With the broad outline of the algorithm in place, we may revisit the question of the closure of \mathcal{F} under conjugation.

Proposition 1 (Closure of \mathcal{F} under conjugation) *Suppose $f \in \mathcal{F}$. Then $f^* \in \mathcal{F}$.*

Proof By the very nature of Fenchel conjugation, it follows that f^* is a closed convex lower semi-continuous function. Furthermore, since f^* is a convex function on the real line it is also upper semi-continuous, and hence continuous. Thus, properties (i)–(iii) are satisfied, and it remains to show that f^* satisfies condition (iv).

By the discussion in Sect. 2.1.3, each $f|_i$ can contribute to f^* by defining it at a single point (iff $f|_i$ is constant) or over an open interval, the end-points of which are defined by the extrema of $f'|_i$ on (x_i, x_{i+1}) . Similarly, each point x_i contributes to f^* by defining it at a single point (iff f is continuous at x_i) or over an open interval, the end-points of which are defined by the the left and right derivatives of f at x_i . Thus, each of the finitely many open intervals (x_i, x_{i+1}) and points x_i of f may contribute to defining f^* at at most one open interval. Thus, f^* is itself defined over finitely many open-intervals. Since f^* is continuous, then each of the constructed $f^*|_i$ are themselves continuous and hence differentiable. Intervals not explicitly constructed by the algorithm are by definition outside of the effective domain of f^* , and defined to take the value ∞ . □

2.1.4 Inversion

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 $f = x^4/4$. 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 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 disappear¹. However, we are

¹ Such functionality has been introduced to version 10 of *Maple*, although in a limited form. The authors are currently working on taking advantage of this new feature to improve the performance of the *SCAT* package.

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 4 ([1], Chap. 3, Theorem 11) *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 $\varepsilon > 0$ is sufficiently small, there exists a corresponding $\delta > 0$ such that for all a with $|a - w_0| < \delta$ the equation $f(z) = a$ has exactly n roots in the disk $|z - z_0| < \varepsilon$.*

Corollary 1 (Location of branch points) *Suppose that f is as in Theorem 4. Suppose furthermore that $f(z)$ is analytic on the entire neighborhood $|z - z_0| < \varepsilon$, and let $g_1(a), \dots, g_n(a)$ represent the n roots of $f(z) = a$ on the neighborhood $|a - w_0| < \delta$. Then $g_1(w_0) = \dots = g_n(w_0) = z_0$.*

Proof Due to the n th order zero of $f(z)$ at z_0 , it follows that $f(z)$ may be expressed as $f(z) - w_0 = (z - z_0)^n g(z)$, where $g(z) \neq 0$, for all z with $|z - z_0| < \varepsilon$. Due to the analyticity of $f(z)$ and the existence of exactly n roots by Theorem 4, for any a with $|a - w_0| < \delta$ we can write $f(z) - a = (z - g_1(a)) \cdots (z - g_n(a))h(z)$, for some $h(z) \neq 0$. Since $\lim_{a \rightarrow w_0} f(z) - a = f(z) - w_0$, it follows that $\lim_{a \rightarrow w_0} (z - g_1(a)) \cdots (z - g_n(a))h(z) = (z - z_0)^n g(z)$, and therefore

$$(z - g_1(w_0)) \cdots (z - g_n(w_0))h(z) = (z - z_0)^n g(z).$$

Suppose $g_i(w_0) \neq z_0$ for some i . Then, since $h(z_0) \neq 0$, it follows that the left hand side of the equation has at most $n - 1$ roots at w_0 , a contradiction. Thus, it must be that $g_1(w_0) = \dots = g_n(w_0) = z_0$. □

As an immediate result of Corollary 1 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 calculate conjugates.

2.2 Many dimensions

The extension of the algorithm to many dimensions relies on the observation that a many dimensional conjugation operation may be realized as a sequence of one-dimensional conjugations. To make this clear, we introduce the concept of a *partial conjugate*. Consider an n -dimensional function $f(\mathbf{x}) = f(x_1, \dots, x_n)$ that has a one-dimensional conjugate calculated with respect to the variable x_i . The notation f^{x_i} then represents the partial conjugate of f with respect to x_i .

Proposition 2 (Iterated Conjugation) Consider a function $f : \mathbb{R}^n \rightarrow \overline{\mathbb{R}}$. Then the conjugate of f may be calculated as an iterated sequence of n conjugation and $n - 1$ negation operations. More formally,

$$f^* = (-(\dots - (f^{x_n} \dots)^{x_2})^{x_1}.$$

Proof The fact can be seen by simply expanding the summation, as follows:

$$\begin{aligned} f^*(\mathbf{y}) &= \sup_{\mathbf{x}} \{ \langle \mathbf{x}, \mathbf{y} \rangle - f(\mathbf{x}) \} \\ &= \sup_{x_1, \dots, x_n} \left\{ \sum_{i=1}^n x_i y_i - f(\mathbf{x}) \right\} \\ &= \sup_{x_1} \left\{ x_1 y_1 + \sup_{x_2} \left\{ x_2 y_2 + \dots + \sup_{x_n} \{ x_n y_n - f(\mathbf{x}) \} \dots \right\} \right\}. \end{aligned}$$

Using the notation of partial conjugates, this may be rewritten as

$$f^* = (-(\dots - (f^{x_n} \dots)^{x_2})^{x_1}.$$

□

2.2.1 A good class of functions

The natural space to work in is the recursive extension to \mathcal{F} . An n dimensional function f is in \mathcal{F}^n if:

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

It follows immediately that \mathcal{F}^n is closed under non-negative scalar multiplication, addition and conjugation.

2.2.2 Fenchel conjugation

Functions in \mathcal{F}^n have an implicit variable order due to their structure. A function defined over the variable order x_1, \dots, x_n may only be partially conjugated along the last variable. 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.

Furthermore, it is important to note that the intermediate partial conjugates are not always well behaved. In particular, they are often not lower semi-continuous in \mathbb{R}^n , although they remain so along the given partial variable. This requires care to be taken in the representation of the intermediate functions to ensure that they are in fact in \mathcal{F}

along the variable we are about to partially conjugate. We illustrate with an example in \mathcal{F}^2 .

Example 1 (Product of roots) Consider the two-dimensional function²

$$f(x_1, x_2) = \begin{cases} \left\{ \begin{array}{l} \infty, \forall x_2, \\ \infty, x_2 < 0 \\ 0, x_2 = 0, \\ 0, 0 < x_2 \end{array} \right. & x_1 < 0 \\ \left\{ \begin{array}{l} \infty, \\ 0, \\ -\sqrt{x_1 x_2}, \end{array} \right. & \begin{array}{l} x_2 < 0 \\ x_2 = 0, \\ 0 < x_2 \end{array} & 0 < x_1 \end{cases}.$$

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 result) yields:

$$f^{x_2}(x_1, y_2) = \begin{cases} \left\{ \begin{array}{l} \infty, \forall y_2, \\ 0, y_2 < 0 \\ 0, y_2 = 0, \\ \infty, 0 < y_2 \end{array} \right. & x_1 < 0 \\ \left\{ \begin{array}{l} \frac{x_1}{4y_2}, \\ \infty, \\ \infty, \end{array} \right. & \begin{array}{l} y_2 < 0 \\ y_2 = 0, \\ 0 < y_2 \end{array} & 0 < x_1 \end{cases}$$

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) .

Close inspection reveals that we have a point of non lower semi-continuity at the origin. As we approach the origin in a straight-line from the point $(x_1, y_2) = (a, b)$ for $a > 0$ and $b < 0$, the function value is constant at $-\infty < \frac{a}{4b} < 0$. Hence, for any neighborhood of the origin,

$$\lim_{(x_1, y_2) \rightarrow (0,0)} \inf f^{x_2}(x_1, y_2) = -\infty < f(0, 0) = 0.$$

Thus, in reordering the variables we must be careful to define the value at the origin that is appropriate for the variable with which we are about to conjugate, preserving continuity. In this example the value at the origin dictated by continuity with respect

² Here and elsewhere in the paper we choose to represent functions in the non-simplified slightly non-standard style used by the *Maple* package SCAT. This representation, while not the most compact, highlights the form of the underlying data-structure representing the function in \mathcal{F}^n .

to x is ∞ . Changing the variable order through inspection, and carefully choosing the value at the origin yields the partial conjugate

$$f^{x_2}(y_2, x_1) = \begin{cases} \left\{ \begin{array}{l} \infty, \quad x_1 < 0 \\ 0, \quad x_1 = 0, \\ \frac{x_1}{4y_2}, \quad 0 < x_1 \end{array} \right. & y_2 < 0 \\ \left\{ \begin{array}{l} \infty, \quad \forall x_1, \end{array} \right. & y_2 = 0 \\ \left\{ \begin{array}{l} \infty, \quad \forall x_1, \end{array} \right. & 0 < y_2 \end{cases}$$

We may now proceed to calculate the complete conjugate by partially conjugating along the x_1 axis. There are two distinct one-dimensional conjugates to be calculated along the line $y_2 = 0$ and the half-plane $y_2 < 0$. This yields:

$$f^*(y_2, y_1) = \begin{cases} \left\{ \begin{array}{l} 0, \quad y_1 < \frac{1}{4y_2} \\ 0, \quad y_1 = \frac{1}{4y_2}, \\ \infty, \quad \frac{1}{4y_2} < y_1 \end{array} \right. & y_2 < 0 \\ \left\{ \begin{array}{l} \infty, \quad \forall y_1, \end{array} \right. & y_2 = 0 \\ \left\{ \begin{array}{l} \infty, \quad \forall y_1, \end{array} \right. & 0 < y_2 \end{cases}$$

It is 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:

$$f^*(y_1, y_2) = \begin{cases} \left\{ \begin{array}{l} 0, \quad y_2 < \frac{1}{4y_1} \\ 0, \quad y_2 = \frac{1}{4y_1}, \\ \infty, \quad \frac{1}{4y_1} < y_2 \end{array} \right. & y_1 < 0 \\ \left\{ \begin{array}{l} \infty, \quad \forall y_2, \end{array} \right. & y_1 = 0 \\ \left\{ \begin{array}{l} \infty, \quad \forall y_2, \end{array} \right. & 0 < y_1 \end{cases}$$

2.2.3 Variable reordering

A function $f \in \mathcal{F}^n$ 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, \dots, x_n , each S_i is naturally represented as $S_i = \{\mathbf{x} : x_1 \in X_1, x_2 \in X_2(x_1), \dots, x_n \in X_n(x_1, \dots, x_{n-1})\}$. The operation of changing variable orders to (for example) x_n, x_1, \dots, x_{n-1} is equivalent to finding \bar{X}_i such that $S_i = \{\mathbf{x} : x_n \in \bar{X}_n, x_1 \in \bar{X}_1(x_n), \dots, x_{n-1} \in \bar{X}_{n-1}(x_n, x_1, \dots, x_{n-2})\}$. 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

$$\int_{S_i} f(\mathbf{x})d\mathbf{x} = \int_{X_1} \dots \int_{X_n} f(\mathbf{x})dx_n \dots dx_1.$$

To change the order of the variables to x_n, x_1, \dots, x_{n-1} we wish to find \bar{X}_i such that:

$$\int_{S_i} f(\mathbf{x}) d\mathbf{x} = \int_{\bar{X}_n} \int_{\bar{X}_1} \cdots \int_{\bar{X}_{n-1}} f(\mathbf{x}) dx_{n-1} \cdots dx_1 dx_n.$$

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 solved. The approach is to first slice the regions $S_i = \{(x_1, x_2) : x_1 \in (a, b), x_2 \in (f(x_1), g(x_2))\}$ at the critical points of f and g in (a, b) . In this fashion we may consider only *monotonic* regions, where the functions f and g are monotone. We may then consider all possible cases where f and g may each be either increasing, constant or decreasing and where one or none of $f(a) = g(a)$ or $f(b) = g(b)$ applies. This analysis leads to 23 distinct sub-problems (further reduced to 12 by symmetry), each of which may be solved assuming appropriate zeroes and inverses may be found [8]. This allows the associated symbolic Fenchel conjugation algorithm to deal quite robustly with non-separable two-dimensional objects, and higher dimensional ones in certain cases.

3 Examples

3.1 The *Maple* Package SCAT

Earlier work by Bauschke and Mohrenschildt [3,4] focussed on symbolically calculating exact subdifferentials and conjugates for one-dimensional real-valued functions on \mathbb{R} , and separable multi-dimensional functions 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 [10,11] 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[InfConv]` 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://ddrive.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 2 (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 := convert(abs(x), PWF);
```

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

We may easily calculate the subdifferential of f1:

```
> sdf1 := SubDiff(f1);
```

$$sdf1 := \begin{cases} \{-1\}, & x < 0 \\ [-1, 1], & x = 0 \\ \{1\}, & x > 0 \end{cases}$$

We may also calculate the conjugate, yielding:

```
> g1 := Conj(f1, y);
```

$$g1 := \begin{cases} \infty, & y < -1 \\ 0, & y = -1 \\ 0, & (-1 < y) \text{ and } (y < 1) \\ 0, & y = 1 \\ \infty, & 1 < y \end{cases}$$

Example 3 (An example from Rockafellar) The following function can be found on page 229 of Rockafellar's text [13]. The function is easily constructed using piecewise and converted to the PWF format:

```
> piecewise(-3 <= x and x <= 1, abs(x) - 2*sqrt(1-x), infinity) :
f5 := convert(%, PWF);
```

$$f5 := \begin{cases} \infty, & x < -3 \\ 1, & x = -3 \\ -2\sqrt{1-x} - x, & (-3 < x) \text{ and } (x < 0) \\ -2, & x = 0 \\ -2\sqrt{1-x} + x, & (0 < x) \text{ and } (x < 1) \\ 1, & x = 1 \\ \infty, & 1 < x \end{cases}$$

We now use the command

```
Plot(f5,x=-4..2,scaling=constrained,axes=framed)
```

to plot the function, yielding Fig. 1. Next, to calculate and plot the subdifferential we use the commands `sdf5 := SubDiff(f5)`, and

```
Plot(sdf5,-3..1,view=[-3..1,-3..5],axes=none),
```

yielding

$$sdf5 := \begin{cases} \{\}, & x < -3 \\ [-\infty, -\frac{1}{2}], & x = -3 \\ \left\{ \frac{(-1+\sqrt{1-x})\sqrt{1-x}}{x-1} \right\}, & (-3 < x) \text{ and } (x < 0) \\ [0, 2], & x = 0 \\ \left\{ -\frac{(1+\sqrt{1-x})\sqrt{1-x}}{x-1} \right\}, & (0 < x) \text{ and } (x < 1) \\ \{\}, & x = 1 \\ \{\}, & 1 < x \end{cases}$$

and the plot in Fig. 2. Finally, we find the conjugate, the biconjugate and manually verify the convexity of `f5` with the following commands:

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

$$g5 := \begin{cases} -3y + 1, & y < -\frac{1}{2} \\ \frac{5}{2}, & y = -\frac{1}{2} \\ \frac{y^2+2y+2}{1+y}, & (-\frac{1}{2} < y) \text{ and } (y < 0) \\ 2, & y = 0 \\ 2, & (0 < y) \text{ and } (y < 2) \\ 2, & y = 2 \\ \frac{y^2-2y+2}{-1+y}, & 2 < y \end{cases}$$

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

```
Equal(f5,F5);
```

true

Fig. 1 Plot of `f5` from Example 3

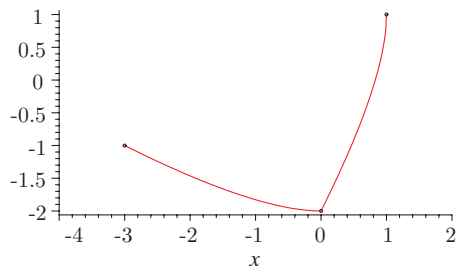
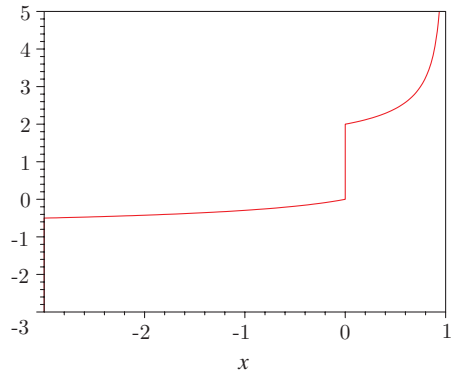


Fig. 2 Plot of `sdf5` from Example 3



Example 4 (Young’s inequality) Suppose $1 < p < \infty$ and let q be such that $\frac{1}{p} + \frac{1}{q} = 1$. The equality

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

is known as *Young’s inequality*. As we are about to see, since $\left(\frac{1}{p}| \cdot |^p\right)^* = \frac{1}{q}| \cdot |^q$ this is actually a special case of the stronger Fenchel–Young inequality from Equation 1. In this example we show and confirm the above conjugate pair.

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);
```

$$f7 := \begin{cases} \frac{(-x)^p}{p}, & x < 0 \\ 0, & x = 0 \\ \frac{x^p}{p}, & 0 < x \end{cases}$$

$$g7 := \begin{cases} \frac{(-\frac{1}{y})^{-q}}{q}, & y < 0 \\ 0, & y = 0 \\ \frac{y^q}{q}, & 0 < y \end{cases}$$

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 Barrier functions

We turn now to the problem of calculating conjugates of some common barrier functions in semi-definite optimization. Many such functions may be generated from appropriate one-dimensional convex functions.

Theorem 5 (Barrier function construction) *Let $\theta : \mathbb{R} \rightarrow \overline{\mathbb{R}}$ be an lower semi-continuous, proper convex and non-decreasing function with $\text{dom } \theta = (-\infty, b)$, $b \in [0, \infty)$, $\lim_{x \rightarrow -\infty} \theta(x) = 0$ and $\lim_{x \rightarrow b} \theta(x) = \infty$. Let $\lambda(D)$ be the vector of eigenvalues of the symmetric matrix $D \in S_n$ in non-decreasing order. Then*

$$H(D) = \sum_{i=1}^n \theta(\lambda_i(D))$$

is a smooth barrier function such that $\lim_{r \rightarrow \infty} r^{-1} H(rD) = \delta_{S_n^-}(D)$, for all $D \in S_n$.

Proof Refer to Propositions 2.6.1, 2.8.1 and Theorem 2.7.2 from [2]. □

A few interesting choices for θ include the following functions taken from [2, p. 74]:

$$\begin{aligned} \theta_1 &= \exp(u), & \text{dom } \theta_1 &= \mathbb{R}; \\ \theta_2 &= -\log(1 - u), & \text{dom } \theta_2 &= (-\infty, 1); \\ \theta_3 &= \frac{u}{1-u}, & \text{dom } \theta_3 &= (-\infty, 1); \\ \theta_4 &= -\log(-u), & \text{dom } \theta_4 &= (-\infty, 0); \text{ and} \\ \theta_5 &= -u^{-1}, & \text{dom } \theta_5 &= (-\infty, 0). \end{aligned}$$

These lead directly to the following barrier functions:

$$\begin{aligned} H_1(D) &= \text{tr}(\exp(D)); \\ H_2(D) &= -\log(\det(I - D)), & \text{for } D \prec I, \infty \text{ otherwise}; \\ H_3(D) &= \text{tr}((I - D)^{-1}D), & \text{for } D \prec I, \infty \text{ otherwise}; \\ H_4(D) &= -\log(\det(-D)), & \text{for } D \prec 0, \infty \text{ otherwise}; \text{ and} \\ H_5(D) &= \text{tr}(-D^{-1}), & \text{for } D \prec 0, \infty \text{ otherwise}. \end{aligned}$$

In all cases, we are able to symbolically calculate the conjugates of the underlying functions θ_i , yielding:

$$\begin{aligned} \theta_1^* &= v(\log(v) - 1), & \text{dom } \theta_1^* &= (0, \infty); \\ \theta_2^* &= v - \log(v) - 1, & \text{dom } \theta_2^* &= (0, \infty); \\ \theta_3^* &= (\sqrt{v} - 1)^2, & \text{dom } \theta_3^* &= [0, \infty); \\ \theta_4^* &= -\log(v) - 1, & \text{dom } \theta_4^* &= (0, \infty); \text{ and} \\ \theta_5^* &= -2\sqrt{v}, & \text{dom } \theta_5^* &= [0, \infty). \end{aligned}$$

Example 5 (Barrier function H_1) Considering the two-dimensional case we find that H_1 reduces to the separable two-dimensional function

$$H_1 = \exp d_1 + \exp d_3, \left(\text{letting } D = \begin{bmatrix} d_1 & d_2 \\ d_2 & d_3 \end{bmatrix} \right),$$

for all $d_1, d_3 \in \mathbb{R}$. The conjugate of this is calculated to be

$$H_1^*(Z) = z_1(\log(z_1) - 1) + z_3(\log(z_3) - 1),$$

for $Z = \begin{bmatrix} z_1 & z_2 \\ z_2 & z_3 \end{bmatrix} \succeq 0$.

Although not powerful enough to calculate conjugates of all the barrier functions H_i for arbitrary dimensions, they do let us explore them in certain lower dimensional cases. We consider a restricted two-dimensional space of symmetric matrices (a slice across the three-dimensional space of matrices S_2)

$$\mathcal{S} = \left\{ \begin{bmatrix} s_1 & s_2 \\ s_2 & s_1 \end{bmatrix} : s_1, s_2 \in \mathbb{R} \right\}.$$

Example 6 (Barrier function H_2) The function H_2 is defined for $D \prec I$. Further restricting ourselves to \mathcal{S} this leads to the domain being the set of matrices $\mathcal{D} = \{D : d_1 < 1, d_1 - 1 \leq d_2 \leq 1 - d_1\}$. Thus, we may define H_2 as

$$H_2 = -\log(1 - 2d_1 + d_1^2 - d_2^2), \quad \text{for } D \in \mathcal{D}.$$

The conjugate is calculated by SCAT as

$$H_2^* := \begin{cases} \begin{matrix} \infty, & \text{all}(z_2), & z_1 < 0 \\ \infty, & \text{all}(z_2), & z_1 = 0 \end{matrix} \\ \begin{matrix} \infty, & z_2 < -z_1 \\ \infty, & z_2 = -z_1 \\ z_1 - \log(z_1 - z_2) - \log(z_1 + z_2) + 2 \log(2) - 2, & (-z_1 < z_2) \text{ and } (z_2 < 0) \\ z_1 - 2 \log(z_1) + 2 \log(2) - 2, & z_2 = 0 \\ z_1 - \log(z_1 - z_2) - \log(z_1 + z_2) + 2 \log(2) - 2, & (0 < z_2) \text{ and } (z_2 < z_1) \end{matrix} \\ \begin{matrix} \infty, & z_2 = z_1 \\ \infty, & z_1 < z_2 \end{matrix} \end{cases}, \quad 0 < z_1$$

With a little massaging we can simplify this to

$$H_2^*(Z) = z_1 - \log(z_1 - z_2) - \log(z_1 + z_2) + 2 \log(2) - 2,$$

for $Z \in \mathcal{S}$ and $Z \succeq 0$.

3.4 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 [6,9]. 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 [9]. 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 [6] 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 $\phi(s) \in L^2_{\mathbb{C}}(D)$, and let $\psi(t) \in L^2_{\mathbb{C}}(D)$ represent the time signal. The problem is that of recovering ϕ from a noisy discrete measurement of ψ , $Y = \psi(\mathbb{J}\delta t) \in \mathbb{C}^m$, where $\mathbb{J} = \{0, \dots, m - 1\}$ and δt represents the sampling interval. The Hoch and Stern entropy is defined by $H(X) = \sum_{j=1}^n h(X_j)$, where h is the convex function on \mathbb{C} given by

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

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

$$(P_{NMR}) \quad \inf \{H(X) : X \in \mathbb{C}^n, \|Y - AX\| \leq \varepsilon\},$$

with the matrix A representing the discrete Fourier transform operator. In [6] the dual of this problem is derived as

$$(D_{NMR}) \quad \sup \{\text{Re}[\langle Y, \Lambda \rangle] - \varepsilon\|\Lambda\| - H^*(A^*\Lambda) : \Lambda \in \mathbb{C}^m\}.$$

The motivation to pursue the dual problem came from the casual observation that $f^*(u^*) = \cosh u^*$ (plotted in Fig. 4), which was calculated by Borwein using SCAT’s

Fig. 3 Plot of $f(|z|)$ from Sect. 3.4

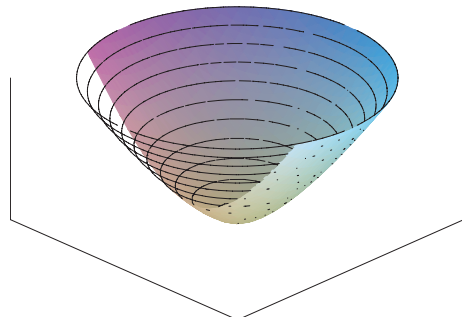
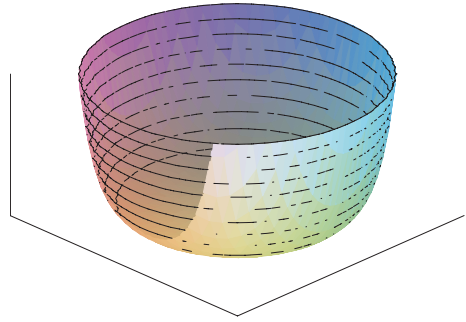


Fig. 4 Plot of $f^*(|z^*|)$ from Sect. 3.4



direct ancestor, `fenchel`. This leads immediately to

$$H^*(X^*) = \sum_{j=1}^n \cosh(b|X_j^*|).$$

Given a solution $\bar{\Lambda}$ to the dual problem, a solution to the primal is found as

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

and a solution to the original spectra as

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

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

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.

Finally, 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 unit ball in \mathbb{R}^n).

5 Concluding remarks

In the *Maple* package *SCAT* we have implemented algorithms for conjugation, sub-differentiation 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.

References

1. Ahlfors, L.: Complex Analysis. McGraw-Hill, New York (1966)
2. Auslender, A., Teboulle, M.: Asymptotic Cones and Functions in Optimization and Variational Inequalities. Springer, Berlin (2003)
3. Bauschke, H., Mohrenschildt, M.v.: Fenchel conjugates and subdifferentiation in Maple. In: Technical Report CORR 97-23, Department of Combinatorics and Optimization, University of Waterloo (1997)
4. Bauschke, H., Mohrenschildt, M.v.: Symbolic computation of Fenchel conjugates. ACM SIGSAM bulletin (2005) (to appear)
5. Borwein, J., Lewis, A.: Convex Analysis and Nonlinear Optimization CMS Books in Mathematics. Springer, New York (2000)
6. Borwein, J., Maréchal, P., Naugler, D.: A convex dual approach to the computation of NMR complex spectra. Math. Meth. Oper. Res. **51**(1), 91–102 (2000). URL:<http://locutus.cs.dal.ca:8088/archive/00000207/>
7. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
8. Hamilton, C.: Symbolic convex analysis. Master's thesis, Department of Mathematics and Statistics, Simon Fraser University (2005)
9. Hoch, J., Stern, A., Donoho, D., Johnstone, I.: Maximum entropy reconstruction of complex (phase-sensitive) spectra. J. Magn. Reson. **86**, 236–246 (1990)
10. Lucet, Y.: A fast computational algorithm for the Legendre–Fenchel transform. Comput. Optim. Appl. **6**(1), 27–57 (1996)
11. Lucet, Y.: Faster than the fast Legendre transform, the linear-time Legendre transform. Numer. Algorithms **16**, 171–185 (1997)
12. Luenberger, D.: Optimization by Vector Space Methods Series in Decision and Control. Wiley, New York (1969)
13. Rockafellar, R.: Convex Analysis. Princeton University Press, Princeton (1970)