Simple implementation and examples for piecewise linearization with abs-normal form

Koichi KUBOTA

Chuo University

AD2016
Outline

**Introduction**
- ANF: Abs-normal form
- rewrite function

**Implementation notes**
- Listing up the abs operations with value zero
- Computation of derivatives
- Local minimization
  - Branch and bound search

**Examples**
- Simple case
- Max-min of many planes
  - Local minimal point $n = 32$ (64 planes, $s = 63$)
  - Not local minimal point $n = 16$ (32 planes, $s = 31$)

**Conclusions and further work**
Previous tools

- Most AD tools can treat the derivative of $|x|$ at $x \neq 0$
- But they can **not** do its derivative at $x = 0$.
  - (The directed derivatives can be computed.)
- Some may display alerts and may compute an element of the subgradient...

Yaad: Yet another AD with piecewise linearization

- computation of piecewise linearization including $\text{abs, min, max}$ at a critical point with abs-normal form (ANF given by A.Griewank[2013])
- reverse/forward accumulation of derivatives
- checking the first order local optimality with Linear Programming
Abs-normal form [Griewank2013]

Absolute Normal Form (ANF) is introduced by A. Griewank (2013). ANF can treat $|x|$ at $x = 0$ systematically by means of the directional derivatives.

Original function and evaluation procedure

$y = f(x)$ \hspace{1cm} ($f : \mathbb{R}^n \rightarrow \mathbb{R}^m$)

\[
\begin{align*}
    w_1 &= \varphi_1(x_1, \cdots, x_n) \\
    w_2 &= \varphi_2(x_1, \cdots, x_n, w_1) \\
    &\vdots \\
    w_k &= \varphi_k(x_1, \cdots, x_n, w_1, \cdots, w_{k-1}) \\
    &\vdots \\
    w_{\ell-m} &= \varphi_{\ell-m}(x_1, \cdots, x_n, w_1, \cdots, w_{\ell-m-1}) \\
    y_1 &= w_{\ell-m+1} = \varphi_{\ell-m+1}(x_1, \cdots, x_n, w_1, \cdots, w_{\ell-m}) \\
    &\vdots \\
    y_m &= w_{\ell} = \varphi_{\ell}(x_1, \cdots, x_n, w_1, \cdots, w_{\ell-m})
\end{align*}
\]
Extract absolute operations with zero arguments

Note that the values of the following $w_{\beta_1}, \ldots, w_{\beta_s}$ are zero!

\[
\begin{align*}
\ldots \\
\varphi_{\alpha_1}(x_1, \ldots, x_n, w_1, \ldots, w_{\alpha_1 - 1}) &= |w_{\beta_1}| \\
\ldots \\
\varphi_{\alpha_2}(x_1, \ldots, x_n, w_1, \ldots, w_{\beta_2 - 1}) &= |w_{\beta_2}| \\
\ldots \\
\varphi_{\alpha_s}(x_1, \ldots, x_n, w_1, \ldots, w_{\beta_s - 1}) &= |w_{\beta_s}| \\
\ldots 
\end{align*}
\]
Rename variables

- results: $w_{\alpha_1} \Rightarrow v_1, \cdots, w_{\alpha_s} \Rightarrow v_s$
- arguments: $w_{\beta_1} \Rightarrow u_1, \cdots, w_{\beta_s} \Rightarrow u_s$

\[
\begin{align*}
v_1 &= |u_1| \\
    & \vdots \\
v_s &= |u_s|
\end{align*}
\]
Define functions: $g : \mathbb{R}^n \times \mathbb{R}^s \to \mathbb{R}^s$, $h : \mathbb{R}^n \times \mathbb{R}^s \to \mathbb{R}^m$

$u = g(x, v)$

$v = \text{abs}(u)$ componentwise: $v_k = \text{abs}(u_k) = |u_k|$

$y = h(x, v)$
Example 1: $f(x_1, x_2) = \min(|x_1 + x_2|, |x_1 - x_2|)$, where $n = 2$, $m = 1$, $\min(a, b) = (a + b - |a - b|) \times 0.5$

Consider the case: $x_1 = 0.0$ and $x_2 = 0.0$ ($s = 3$)

$w_1 = x_1 + x_2$
$w_2 = \text{abs}(w_1)$
$w_3 = x_1 - x_2$
$w_4 = \text{abs}(w_3)$
$w_5 = w_2 - w_4$
$w_6 = \text{abs}(w_5)$
$w_7 = w_2 + w_4 - w_6$
y = $w_8 = w_7 \times 0.5$

$u_1 \equiv w_1,$ $u_2 \equiv w_3,$ $u_3 \equiv w_5,$
$v_1 \equiv w_2 = |u_1|,$ $v_2 \equiv w_4 = |u_2|,$ $v_3 \equiv w_6 = |u_3|.$
Example 1 (mod): \( f(x_1, x_2) = \min(|x_1 + x_2|, |x_1 - x_2|) \), where \( n = 2, m = 1, \min(a, b) = (a + b - |a - b|) \times 0.5 \)

Consider the case: \( x_1 = 0.1 \) and \( x_2 = 0.1 \) (\( s = 1 \))

\[
\begin{align*}
  w_1 & = x_1 + x_2 \\
  w_2 & = \text{abs}(w_1) \\
  w_3 & = x_1 - x_2 \\
  w_4 & = \text{abs}(w_3) \\
  w_5 & = w_2 - w_4 \\
  w_6 & = \text{abs}(w_5) \\
  w_7 & = w_2 + w_4 - w_6 \\
  y & = w_8 = w_7 \times 0.5
\end{align*}
\]

\[
\begin{align*}
  u_1 & \equiv w_3, \\
  v_1 & \equiv w_4 = |u_1|.
\end{align*}
\]
Rewrite the function

**Example 1: \( x_1 = 0.0 \) and \( x_2 = 0.0 \)**

\[
\begin{align*}
  u_1 &= x_1 + x_2 & u_1 &= x_1 + x_2 \quad ([1]) \\
  v_1 &= \text{abs}(u_1) & u_2 &= x_1 - x_2 \quad ([3]) \\
  u_2 &= x_1 - x_2 & u_3 &= v_1 - v_2 \quad ([5]) \\
  v_2 &= \text{abs}(u_2) & v_1 &= \text{abs}(u_1) \quad ([2]) \\
  u_3 &= v_1 - v_2 & v_2 &= \text{abs}(u_2) \quad ([4]) \\
  v_3 &= \text{abs}(u_3) & v_3 &= \text{abs}(u_3) \quad ([6]) \\
  w_7 &= v_1 + v_2 - v_3 & w_7 &= v_1 + v_2 - v_3 \quad ([7]) \\
  y &= w_8 = w_7 \times 0.5 & y &= w_8 = w_7 \times 0.5 \quad ([8])
\end{align*}
\]

\[
\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \mathbf{g}(x_1, x_2, v_1, v_2, v_3) = \begin{pmatrix} x_1 + x_2 \\ x_1 - x_2 \\ v_1 - v_2 \end{pmatrix} \\
\]

\[
y = \mathbf{h}(x_1, x_2, v_1, v_2, v_3) = 0.5v_1 + 0.5v_2 - 0.5v_3
\]
Note that $u_k = 0.0$ for $k = 1, 2, 3$.

**rewrite for linearization**

$$
\begin{align*}
\Delta u_1 &= \Delta x_1 + \Delta x_2 \\
\Delta v_1 &= \text{abs}(\Delta u_1) = \text{sign}(\Delta u_1) \cdot \Delta u_1 = \sigma_1 \Delta u_1 \\
\Delta u_2 &= \Delta x_1 - \Delta x_2 \\
\Delta v_2 &= \text{abs}(\Delta u_2) = \sigma_2 \Delta u_2 \\
\Delta u_3 &= \Delta v_1 - \Delta v_2 \\
\Delta v_3 &= \text{abs}(\Delta u_3) = \sigma_3 \Delta u_3 \\
\Delta w_7 &= \Delta v_1 + \Delta v_2 - \Delta v_3 \\
\Delta y &= \Delta w_8 &= \Delta w_7 \times 0.5 \\
\sigma_k &= \begin{cases} 
1 & \text{if } \Delta u_k > 0 \\
-1 & \text{if } \Delta u_k < 0 
\end{cases}
\end{align*}
$$
Rewrite procedure for linearization

Note that $u_k = 0.0$ for $k = 1, 2, 3$.

**rewrite for linearization**

\[
\begin{align*}
\Delta u_1 &= \Delta x_1 + \Delta x_2 \\
\Delta u_2 &= \Delta x_1 - \Delta x_2 \\
\Delta u_3 &= \Delta v_1 - \Delta v_2 \\
\Delta v_1 &= \text{abs}(\Delta u_1) = \sigma_1 \Delta u_1 \\
\Delta v_2 &= \text{abs}(\Delta u_2) = \sigma_2 \Delta u_2 \\
\Delta v_3 &= \text{abs}(\Delta u_3) = \sigma_3 \Delta u_3 \\
\Delta w_7 &= \Delta v_1 + \Delta v_2 - \Delta v_3 \\
\Delta y &= \Delta w_8 = \Delta w_7 \times 0.5
\end{align*}
\]

\[
\sigma_k = \begin{cases} 
1 & \text{if } \Delta u_k > 0 \\
-1 & \text{if } \Delta u_k < 0 
\end{cases}
\]
Differentiation of Absolute function [Griewank2013]

Absolute function

\[ v = |u| = \text{abs}(u) = \text{sign}(u) \cdot u \]

\[ v + \Delta v = |u + \Delta u| \]
\[ = \text{sign}(u + \Delta u) \cdot (u + \Delta u) \]
\[ = \text{sign}(u + \Delta u) \cdot \Delta u + \text{sign}(u + \Delta u) \cdot u \]

\[ \Delta v = \text{sign}(u + \Delta u) \cdot \Delta u + (\text{sign}(u + \Delta u) - \text{sign}(u)) \cdot u \]

when \( u = 0 \)

\[ \Delta v = \text{sign}(\Delta u) \cdot \Delta u = |\Delta u| \]
Now we can compute the directed derivatives for given $\Delta x$:

$$y = f(x)$$

$$y + \Delta y = f(x + \Delta x) = f(x) + \frac{\partial f}{\partial x} \Delta x + \cdots$$

rewrite the above as follows:

$$\Delta u = \frac{\partial g}{\partial x} \Delta x + \frac{\partial g}{\partial v} \Delta v,$$

$$\Delta v = \text{sign}(u + \Delta u) \cdot \Delta u + (\text{sign}(u + \Delta u) - \text{sign}(u)) \cdot u,$$

$$= |\Delta u| \text{ when all elements of } u \text{ are zeros}$$

$$\Delta y = \frac{\partial h}{\partial x} \Delta x + \frac{\partial h}{\partial v} \Delta v.$$

Note that when we give a value of direction $\Delta x$, we get the values of $\Delta u$, $\Delta v$ and $\Delta y$. Moreover, when we assume the sign of $\Delta u$ firstly, we get the coefficients of $\Delta v$ and $\Delta y$ with respect to linear combinations of $\Delta x = (\Delta x_1, \ldots, \Delta x_n)$. 
We check the above concepts with C++ operator overload program for generating computational graph $G$.

For a scalar function $f(x)$, we check the first order optimality condition (minimization or maximization) by solving LP (Linear Programming) repeatedly.

- The number of solving different LP is $O(2^s)$ (naively count)
- Reduce the number by branch and bound.
Computation of derivatives

implementation outline

The partial derivatives $\frac{\partial g}{\partial x}$, $\frac{\partial g}{\partial v}$, $\frac{\partial h}{\partial x}$, $\frac{\partial h}{\partial v}$ are computed as follows.

(i) Make a computational graph $G$ of $f$ as well as the sets of renamed nodes, i.e., $V = \{v_1, \ldots, v_s\}$ and $U = \{u_1, \ldots, u_s\}$.

(ii) For each node $v$ in $V$, change the node type of $v$ from the type that is the result of abs() operation to the type that is the new independent variable.

(iii) Compute $\frac{\partial g_k}{\partial x_j}$ ($j = 1, \ldots, n$) and $\frac{\partial g_k}{\partial v_\ell}$ ($\ell = 1, \ldots, s$) by reverse mode (or forward mode) for all $u_k \in U$ ($k = 1, \ldots, s$). That is, after the topological sort with the depth first search from $u_k$, the adjoint values are computed for all $w < u_k \in G$.

(iv) Compute $\frac{\partial h}{\partial x_j}$ ($j = 1, \ldots, n$) and $\frac{\partial h}{\partial v_\ell}$ ($\ell = 1, \ldots, s$) by reverse mode from the node corresponding to $y = f(x)$. 
$\Delta u = (\partial g/\partial x) \Delta x + (\partial g/\partial v) \Delta v$

$\Delta v = \Sigma \Delta u,$

where $\Sigma \equiv \text{diag}(\sigma_1, \ldots, \sigma_s)$ is a diagonal matrix.

We can eliminate $\Delta v$ and get the explicit form $\Delta u$ with respect to $\Delta x$, that is,

$\Delta u = (I - (\partial g/\partial v) \Sigma)^{-1} (\partial g/\partial x) \Delta x.$

Thus the directed derivative $\Delta y = (\partial f/\partial x) \Delta x$ is computed by

$\Delta y = (\partial h/\partial x) \Delta x + (\partial h/\partial v) \sum (I - (\partial g/\partial v) \Sigma)^{-1} (\partial g/\partial x) \Delta x. \quad (1)$

$U \equiv (\partial g/\partial x), \ L \equiv (\partial g/\partial v), \ J \equiv (\partial h/\partial x)$ and $V \equiv (\partial h/\partial v)$.

$\Delta y = (J + V \Sigma (I - L \Sigma)^{-1} U) \Delta x$
Check the local minimum of the linearization

One of key advantages of the abs-normal form is that the coefficients of a directional derivative $\Delta y$ can be computed with the sign of the $\Delta u$. The sign of the $\Delta u$ is computed by the value of the direction $\Delta x$.

**Usually**
Given $\Delta x$, compute $y = f(x) \Rightarrow y + \Delta y = f(x + \Delta x)$.

**Usually (Abs-normal form)**
Given $\Delta x$, compute $u = g(x, v)$, $v = |u| \Rightarrow y = h(x, v)$
$\Sigma$ is determined by $\Delta x$
$\Delta u = U\Delta x, \Delta v = \Sigma\Delta u \Rightarrow \Delta y = J\Delta x + V\Delta v$

**Fix $\Sigma$ first (Abs-normal form)**
Fix $\Sigma$, compute $\Delta y = J\Delta x + V\Delta v$
Check the existence of the direction $\Delta x$ that realizes the given $\Sigma$. 
The $k$-th diagonal element of $\Sigma$, $\sigma_k$, indicates the sign of $\Delta u_k$.

**Check the existence (or finding the subdomain)**

- Compute $\Delta u_k$ as a linear combination of $\Delta x$ $(k = 1, \cdots, s)$
  - $\Delta u_k > 0$ for $\sigma_k = 1$, or, $\Delta u_k < 0$ for $\sigma_k = -1$
- Construct an $s \times n$ matrix $A$ whose $k$-th row is $\Delta u_k$ (if $\sigma_k = 1$) or $-\Delta u_k$ (if $\sigma_k = -1$)
- Check the feasibility of an LP:
  \[
  \min_{\Delta x} \Delta y \text{ s.t. } A\Delta x > 0 \text{ and } -\infty < \Delta x_j < \infty
  \]
  - if it is feasible, there is a direction $\Delta x$ that realizes all the sign of $\Delta u_k$’s equal to $\sigma_k$’s.
  - if it is not feasible, there is no direction that realizes $\Sigma$. 
Check the first order condition of locally minimum of the linearization

There are $2^s$ combinations of the values of $\sigma_k$'s.

Thus, we can compute a direction that gives $\Delta y < 0$ (or $\Delta y > 0$) after solving LP $2^s$ times.

When we want to know locally minimum (maximum), we check there are no direction that gives $\Delta y < 0$ ($\Delta y > 0$).
(i) Fix a combination of the diagonal values of $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_s) = \text{diag}(\pm 1, \ldots, \pm 1)$.

(ii) Compute $\Delta u = (I - L\Sigma)^{-1} U \Delta x$ and $\Delta y = (J + V\Sigma(I - L\Sigma)^{-1}U) \Delta x$. The coefficients of $\Delta x_1, \ldots, \Delta x_n$ of the above $\Delta y$ may give one of the generalized gradients.

(iii) Check the direction $\Delta x$ that gives the directed derivative $\Delta y$ by feasibility of $A\Delta x > 0$ with LP.

(iv) Finally, for each matrix $A$ corresponding to all the possible diagonal matrices $\Sigma$, we can check the infeasibility of $\Delta y < 0$ and $A\Delta x > 0$. When there are no feasible solutions the given point $x$ is local minimum or stationary point of the linearization.
Branch and bound search

- The number of solving LP is at most $2^s$.
- Let a $k \times n$ matrix $\bar{A}$ denote submatrix of $A$ ($k \leq s$).
  - When $A\Delta x > 0$ is feasible, any subsystem of $\bar{A}\Delta x > 0$ should be feasible,
  - When $\bar{A}\Delta x > 0$ is infeasible, its extended system $A\Delta x > 0$ should be infeasible.
- Construct matrix $A$ from the first row to $s$th row with step by step ($g_k$ indicates the $k$-th component of $g(x, v)$)

$$
\Delta u_1 = \frac{\partial g_1}{\partial x} \Delta x
$$

$$
\Delta u_2 = \frac{\partial g_2}{\partial x} \Delta x + \frac{\partial g_2}{\partial v_1} \Delta v_1 = \frac{\partial g_2}{\partial x} \Delta x + \frac{\partial g_2}{\partial v_1} \sigma_1 \Delta u_1
$$

\[ \vdots \]

$$
\Delta u_k = \frac{\partial g_k}{\partial x} \Delta x + \sum_{j=1}^{k-1} \frac{\partial g_k}{\partial v_j} \sigma_j \Delta u_j
$$
Construct sub matrix $A^{(k)}$

$$
\Delta u_k = \frac{\partial g_k}{\partial x} \Delta x + \sum_{j=1}^{k-1} \frac{\partial g_k}{\partial v_j} \cdot \sigma_j \cdot \Delta u_j \quad (k = 1, \ldots, s).
$$

After the computation of $\Delta u_k$ in the explicit form with respect to $\Delta x$ under the selected sign combination of $\sigma_1, \ldots, \sigma_k$, we have $\Delta u_1, \ldots, \Delta u_k$.

Then, we can check the feasibility of $A^{(k)} \Delta x > 0$, where the coefficient matrix is defined by $A^{(k)} \equiv \begin{pmatrix} 
\sigma_1 \Delta u_1 \\
\vdots \\
\sigma_k \Delta u_k 
\end{pmatrix}$.

- When $A^{(k)} \Delta x > 0$ is feasible, we should check $A^{(k+1)} \Delta x > 0$ is feasible or infeasible for $\sigma_{k+1} = 1$ and $\sigma_{k+1} = -1$.
- When it is infeasible, there are no feasible solution with the current $\sigma_1, \ldots, \sigma_k$. Try the next combination.
Example

Again, \( f(x_1, x_2) = \min(|x_1 + x_2|, |x_1 - x_2|) \), \( x_1 = 0.0 \) and \( x_2 = 0.0 \).

**Abs-normal form**

\[
\Delta u = \begin{pmatrix}
\Delta u_1 \\
\Delta u_2 \\
\Delta u_3
\end{pmatrix} = \begin{pmatrix}
\Delta x_1 + \Delta x_2 \\
\Delta x_1 - \Delta x_2 \\
\Delta v_1 - \Delta v_2
\end{pmatrix}, \quad \begin{pmatrix}
\Delta v_1 \\
\Delta v_2 \\
\Delta v_3
\end{pmatrix} = \begin{pmatrix}
\sigma_1 \Delta u_1 \\
\sigma_2 \Delta u_2 \\
\sigma_3 \Delta u_3
\end{pmatrix},
\]

\( \Delta y = 0.5 * (\Delta v_1 + \Delta v_2 - \Delta v_3) \)

**Partial derivatives**

With the forward AD technique, we have

\[
U = (\partial g / \partial x) = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ 0 & 0 \end{pmatrix}, \quad L = (\partial g / \partial v) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & -1 & 0 \end{pmatrix},
\]

\[
J = (\partial h / \partial x) = ( 0 \ 0 ), \quad V = (\partial h / \partial v) = ( 0.5 \ 0.5 \ -0.5 ).
\]
Explicit $\Delta y$ form with respect to $\Delta x$

**Case:** $\Delta u_1 > 0$, $\Delta u_2 > 0$ and $\Delta u_3 > 0$

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \Delta v_1 = \Delta u_1, \Delta v_2 = \Delta u_2, \Delta v_3 = \Delta u_3.$$

The conditions, $\Delta u_1 = \Delta x_1 + \Delta x_2 > 0$, $\Delta u_2 = \Delta x_1 - \Delta x_2 > 0$ and $\Delta u_3 = \Delta v_1 - \Delta v_2 = \Delta u_1 - \Delta u_2 = 2\Delta x_2 > 0$, hold for $\Delta x_1 > \Delta x_2 > 0$.

$$\Delta y = (J + V\Sigma(I - L\Sigma)^{-1}U)\Delta x$$

$$= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ 0 & 0 \end{pmatrix} \Delta x$$

$$= \begin{pmatrix} 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ 1 & -1 \end{pmatrix} \Delta x = \Delta x_1 - \Delta x_2.$$
Figures

![Graph 1: 3D plot of min(abs(y-x), abs(y+x))](image)

![Graph 2: 2D plot with axes Δx₁ and Δx₂](image)
### Generalized gradient

#### Table: The generalized gradient $\Delta y$

<table>
<thead>
<tr>
<th>No.</th>
<th>$\Delta u_1$</th>
<th>$\Delta u_2$</th>
<th>$\Delta u_3$</th>
<th>$\sigma_1$</th>
<th>$\sigma_2$</th>
<th>$\sigma_3$</th>
<th>Domain</th>
<th>$\Delta y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>$&gt; 0$</td>
<td>$&gt; 0$</td>
<td>$&gt; 0$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\Delta x_1 &gt; \Delta x_2 &gt; 0$</td>
<td>$\Delta x_1 - \Delta x_2$</td>
</tr>
<tr>
<td>(2)</td>
<td>$&lt; 0$</td>
<td>$&gt; 0$</td>
<td>$&gt; 0$</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>$0 &gt; \Delta x_1 &gt; \Delta x_2$</td>
<td>$\Delta x_1 - \Delta x_2$</td>
</tr>
<tr>
<td>(3)</td>
<td>$&gt; 0$</td>
<td>$&lt; 0$</td>
<td>$&gt; 0$</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>$0 &lt; \Delta x_1 &lt; \Delta x_2$</td>
<td>$-\Delta x_1 + \Delta x_2$</td>
</tr>
<tr>
<td>(4)</td>
<td>$&lt; 0$</td>
<td>$&lt; 0$</td>
<td>$&gt; 0$</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>$\Delta x_1 &lt; \Delta x_2 &lt; 0$</td>
<td>$-\Delta x_1 + \Delta x_2$</td>
</tr>
<tr>
<td>(5)</td>
<td>$&gt; 0$</td>
<td>$&gt; 0$</td>
<td>$&lt; 0$</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>$\Delta x_1 &gt; -\Delta x_2 &gt; 0$</td>
<td>$\Delta x_1 + \Delta x_2$</td>
</tr>
<tr>
<td>(6)</td>
<td>$&lt; 0$</td>
<td>$&gt; 0$</td>
<td>$&lt; 0$</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
<td>$0 &lt; \Delta x_1 &lt; -\Delta x_2$</td>
<td>$-\Delta x_1 - \Delta x_2$</td>
</tr>
<tr>
<td>(7)</td>
<td>$&gt; 0$</td>
<td>$&lt; 0$</td>
<td>$&lt; 0$</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>$0 &lt; -\Delta x_1 &lt; \Delta x_2$</td>
<td>$\Delta x_1 + \Delta x_2$</td>
</tr>
<tr>
<td>(8)</td>
<td>$&lt; 0$</td>
<td>$&lt; 0$</td>
<td>$&lt; 0$</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>$-\Delta x_1 &gt; \Delta x_2 &gt; 0$</td>
<td>$-\Delta x_1 - \Delta x_2$</td>
</tr>
</tbody>
</table>
Max-min of planes

Scalar function $f(x_1, x_2)$ is defined with $2n$ planes as

$$f(x_1, x_2) \equiv \max_{0 \leq \ell < n} \min(a_{2\ell}x_1 + b_{2\ell}x_2, a_{2\ell+1}x_1 + b_{2\ell+1}x_2).$$

The $k$th plane $a_kx_1 + b_kx_2$ ($k = 0, \ldots, 2n - 1$) is defined by three points $(p_k, q_k, r_k), (p_{k+1}, q_{k+1}, r_{k+1})$ and $(0, 0, 0)$, where $(p_k, q_k) = (\cos(\frac{\pi}{n}k), \sin(\frac{\pi}{n}k))$ and arbitrarily given $r_k$ $(p_{2n}, q_{2n}, r_{2n}) \equiv (p_0, q_0, r_0)$.

**Figure**: Origin is local minimal point
$n = 32$, $r_{2k} = 0.3$, $r_{2k+1} = 1.0$ ($k = 0, \ldots, 31$).
Locally minimal point at $(0, 0)$.
There are 63 absolute operations whose results are zero, $s = 63$.
The number of solving LP is $2^s = 2^{63} \approx 10^{19}$
the total number of solving LP is only $10^{340}$ with the branch and bound search.
Computational time is about 7 seconds (Ubuntu 14.04LTS, VMware Fusion 7.1.3, Macbook pro core i7).

**Figure:** Origin is local minimal point
Example 2-2

\[ n = 16, \quad r_{26} = -0.1, \quad r_{2k} = 0.3 \ (k = 0, \ldots, 12, 14, 15), \text{ and } \]
\[ r_{2k+1} = 1.0 \ (k = 0, \ldots, 15). \]
The origin (0, 0) is not locally minimal.
There are directions along with which the value of \( f \) is decreased.
There are 31 absolute operations whose results are zero, \( s = 31 \).
The number of solving LP is \( 2^s = 2^{31} \approx 2 \times 10^9 \)
the total number of solving LP is only 2222 with the branch and bound search.

**Figure**: Origin is not local minimal point
Conclusion:

- Simple implementation of piecewise linearization with “abs-normal-form” in C++
- An efficient way to check the local minimal point with branch and bound technique,
- Two dimensional examples that may have $2n$ planes were shown, where the number of solving LP is reduced: $2^{63} \rightarrow 10340$, $2^{31} \rightarrow 2222$.

Future work:

- More practical experiments and the investigation of the optimal topological order for the branch and bound are needed
- the investigation of the higher derivatives with absolute operations and effects of numerical computational errors.