

IMA MINLP Workshop, Nov. 2008

Using Expression Graphs in Optimization Algorithms

David M. Gay

Optimization and Uncertainty Estimation

<http://www.sandia.gov/~dmgay>

dmgay@sandia.gov

+1-505-284-1456

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Released as SAND2008-7541C.



Outline

- The Problem
- Some algorithm ingredients
- Expression graphs
 - various forms
 - derivative computations
 - bound computations
 - convexity detection
- Summary and pointers



The Problem

Seek x^* to minimize $f(x)$

$$\text{s.t. } \underline{\ell} \leq c(x) \leq \underline{u}$$

$$f : \mathcal{R}^n \rightarrow \mathcal{R}$$

$$c : \mathcal{R}^n \rightarrow \mathcal{R}^m$$

with f, c smooth; $x \in D$ compact.

Settle for \hat{x} with $f(\hat{x}) - f(x^*) < \epsilon$.

For MINLP, $n = p + q$, $D = \mathcal{R}^p \times \mathcal{Z}^q$.



Branch & Bound, Reduce, Cut...

Algorithm ingredients... in parallel,

- Local search for good *incumbents*.
- Compute bounds (e.g., relax).
- Compute, refine convex outer approx's.
- Test for sufficiency, exclusion.
- Branch — split domain as needed.
- Reduce domain (using convexity).
- Presolve, updated with new info.

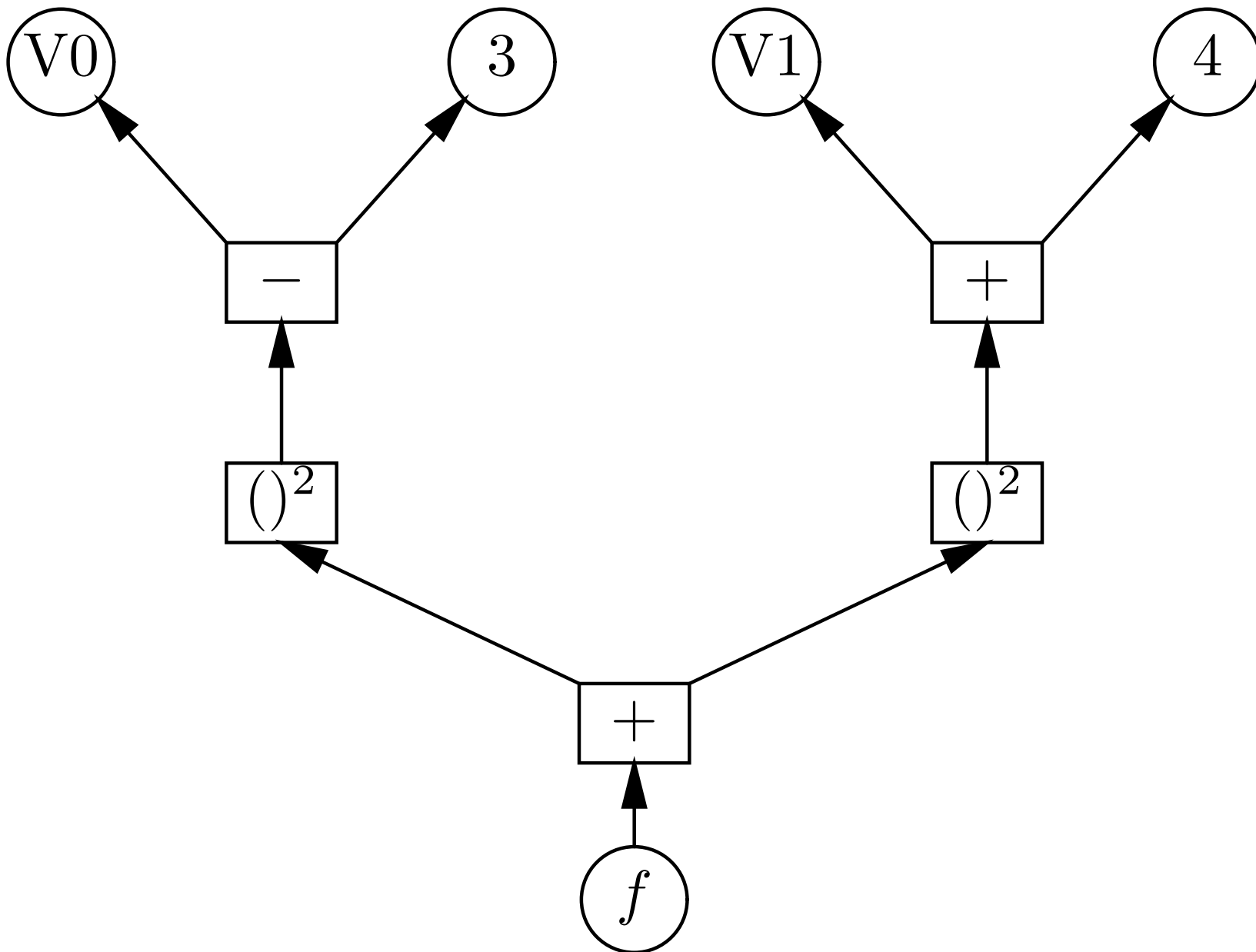


Expression Graphs

Good for

- Function evaluations (doing, simplifying)
- Derivative computations (AD)
- Bound computations, e.g.,
 - interval
 - Taylor series
 - slope
- Convexity detection.

Expression Graph for $f = (x - 3)^2 + (y + 4)^2$





Expression Graph Representations

Possible representations include

- graph = list-style data structures
- list of tuples
- Polish prefix or postfix
- XML

Convert from one to another in linear time.

AMPL/solver interface lib. (ASL) uses Polish prefix for external rep., graph for internal.



ASL Graph Walks

Graph walks in ASL currently include...

- conversion to internal form with AD setup
- detection of quadratic forms
- detection of partially-separable structure for efficient Hessian computations
- operator adjustments (for evaluations after some of the above)



Example: ASL Multiplication Operator

Do multiplication and save partials for AD:

```
double
f_OPMULT(expr *e A_AS L)
{
    expr *e1 = e->L.e;
    expr *e2 = e->R.e;
    return    (e->dR = (*e1->op)(e1))
              * (e->dL = (*e2->op)(e2));
}
```



Forward AD via Graph Walk

Computing $x_j = o_j(x_k, x_\ell)$ (with $j > n$, $k < j$, $\ell < j$)

$$\implies \frac{\partial x_j}{\partial x_i} = \frac{\partial o_j}{\partial x_k} \frac{\partial x_k}{\partial x_i} + \frac{\partial o_j}{\partial x_\ell} \frac{\partial x_\ell}{\partial x_i} \quad \text{for } 1 \leq i \leq n.$$

Similarly recur higher derivatives by graph walk doing forward AD.



Partially Separable Structure

$$f(x) = \sum_i f_i(A_i x)$$

$$\implies \nabla^2 f(x) = \sum_i A_i^T \nabla^2 f_i A_i$$

Graph walk finds “group” partial separability:

$$f(x) = \sum_i \theta_i \left(\sum_j^{r_i} f_{ij}(A_{ij} x) \right)$$



Use of Partially Separable Structure

Good for efficiently computing explicit Hessians and Hessian-vector products.

In ASL, partials are stored during function evaluations (graph walks) for use in Hessian-vector computations by a mix of forward and backward AD.



More Computations by Graph Walks

Walks similar to forward AD can compute

- interval bounds (by interval arithmetic);
- propagate Taylor series;
- compute interval slopes.

“Slopes” are divided differences:

$$f[x, z] = \begin{cases} (f(x) - f(z)) / (x - z) & \text{if } x \neq z \\ f'(x) & \text{if } x = z \end{cases}$$

Slope Arithmetic

Slope arithmetic, analogous to forward AD
[Krawczyk & Neumaier, 1985]:

$$\begin{array}{l} \underline{f = \dots} \quad \Rightarrow \quad \underline{f[x, z] = \dots} \\ c \in \mathcal{R} \quad 0 \\ x \quad 1 \\ g \pm h \quad g[x, z] \pm h[x, z] \\ g \cdot h \quad g[x, z] \cdot h(x) + g(z) \cdot h[x, z] \\ g/h \quad (g[x, z] - h[x, z] \cdot f(z))/h(x) \end{array}$$



Interval Slopes

Interval X , interval evaluation of $f[X, z]$

$$\Rightarrow f[x, z] \in f[X, z] \quad \forall x \in X.$$

$$f(x) = f(z) + f[x, z](x - z)$$

$$\Rightarrow f(X) \subseteq F_z(X) \doteq f(z) + f[X, z](X - z).$$

Quadratic approximation:

$$\text{width}(F_z(X)) - \text{width}(f(X)) \leq O(\text{width}(X)^2).$$



Extension to n Variables

Interval slope computations extend readily to n variables and can be done by walk of expression graph.

$$\mathit{work}(f[X, z]) = O(n \cdot \mathit{work}(f(x))).$$



Second-Order Slopes

Second-order slopes:

- Not unique.
- $f(x) =$
 $f(z) + f'(z)(x - z) + (x - z)^T f[x, z, z](x - z).$
- $(x - z)^T f[x, z, z] = f[x, z] - f'(z).$
- For $x \in \mathcal{R}^n$,
 $work(f[X, z, z]) = O(n^2 \cdot work(f(x))).$



Convexity

Can specify some problems in a way that guarantees convexity, e.g.,

- CVXMOD [Boyd & Mattingley, 2006]
- Joseph Young thesis (Rice, 2008)

but in general have convexity only in some regions.



Convexity Detection

Can walk expression graphs to detect condition sufficient for convexity in a region.

- [Nenov, Fylstra, Kolev, 2004], in Frontline spreadsheet software
- Dr. AMPL [Fourer, et al., 2008]



Summary

With walks of expression graphs we can

- Evaluate expressions
- Detect problem structure:
 - convexity, partially separable structure.
- Compute derivatives
- Compute bounds
- Presolve (linear, nonlinear)
- Optimize such computations.



Pointers

- Neumaier global optimization page/pointers:
http:
`//www.mat.univie.ac.at/~neum/glopt.html`
- Coconut (C++ software): http:
`//www.mat.univie.ac.at/~neum/glopt/coconut`
- AMPL web site (e.g., some papers):
`http://www.ampl.com`
- My papers: pointers in
`http://www.cs.sandia.gov/~dmgay`