Multiplication on affine forms using Semidefinite Programming

MeASI

September 15, 2008

We use affine forms [[**?**]] to represent variables. Let x be a variable within an interval $\mathbf{I}$ (assumed to have finite bounds), its related affine form, $\hat{x}$, is as follow.

$$\hat{x} = \alpha_0^x + \sum_{i=1}^{n} \alpha_i^x \epsilon_i,$$

where noise symbols, ϵ_i, are assumed to have their unknown values within $[-1, 1]$. Properties below are always satisfied.

$$\forall x \in \mathbf{I}, \exists(\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n) \in [-1,1]^n | x = \alpha_0^x + \sum_{i=1}^{n} \alpha_i^x \varepsilon_i$$

$$\forall(\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n) \in [-1,1]^n, \alpha_0^x + \sum_{i=1}^{n} \alpha_i^x \varepsilon_i \in \mathbf{I}.$$

Multiplication operation . We denote by $\mathcal{A}$ the set of affine forms which can be seen as a direct sum of $\mathbb{R}$ and $\mathbb{R}^n$, that is, $\mathcal{A} = \mathbb{R} \oplus \mathbb{R}^n$. A variable x in $\mathcal{A}$ is defined by a constant, α_0^x, in $\mathbb{R}$ plus a vector, $V^x := (\alpha_1^x, .., \alpha_n^x)^t$ in $\mathbb{R}^n$.

Let x and y be two variables lying on I_x and I_y respectively, let $\hat{x}$ and $\hat{y}$ be their respective affine forms. Multiplying two affine forms produces a non linear term.

$$\begin{aligned} \hat{z} \quad = \hat{x} * \hat{y} \quad &= (\alpha_0^x + \textstyle\sum_{i=1}^{n} \alpha_i^x \epsilon_i) * (\alpha_0^y + \sum_{i=1}^{n} \alpha_i^y \epsilon_i) \\ &= \alpha_0^x \alpha_0^y + \textstyle\sum_{i=1}^{n} (\alpha_i^x \alpha_0^y + \alpha_i^y \alpha_0^x)\epsilon_i + \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i^x \alpha_j^y \epsilon_i \epsilon_j \end{aligned}$$

The affine form, $\hat{z}$, can be seen as a function of ϵ_i, parameterized by coefficients of $\hat{x}$ and $\hat{y}$. It is in fact a quadratic form and its related matrix is the kronecker product of V^x and V^y:

$$q(t) = t.Q_{x,y}.t^t$$

Where,

$$t = (1, \epsilon_1, \epsilon_2, \ldots, \epsilon_n)$$

and,

$$Q_{x,y} = (\alpha_0^x, \alpha_1^x, \ldots, \alpha_n^x) \otimes \begin{pmatrix} \alpha_0^y \\ \alpha_1^y \\ \vdots \\ \alpha_n^y \end{pmatrix} = \begin{pmatrix} \alpha_0^x\alpha_0^y & \alpha_1^x\alpha_0^y & \ldots & \alpha_n^x\alpha_0^y \\ \alpha_0^x\alpha_1^y & \alpha_1^x\alpha_1^y & \ldots & \alpha_n^x\alpha_1^y \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_0^x\alpha_n^y & \alpha_1^x\alpha_n^y & \ldots & \alpha_n^x\alpha_n^y \end{pmatrix}$$

BLAS: The BLAS (Basic Linear Algebra Subprograms) are routines that provide standard building blocks for performing basic vector and matrix operations. The Level 1 BLAS perform scalar, vector and vector-vector operations, the Level 2 BLAS perform matrix-vector operations, and the Level 3 BLAS perform matrix-matrix operations. Because the BLAS are efficient, portable, and widely available, they are commonly used in the development of high quality linear algebra software, LAPACK for example.

LAPACK: LAPACK is written in Fortran77 and provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems. The associated matrix factorizations (LU, Cholesky, QR, SVD, Schur, generalized Schur) are also provided, as are related computations such as reordering of the Schur factorizations and estimating condition numbers. Dense and banded matrices are handled, but not general sparse matrices. In all areas, similar functionality is provided for real and complex matrices, in both single and double precision.

CLAPACK: The CLAPACK library was built using a Fortran to C conversion utility called f2c. The entire Fortran 77 LAPACK library is run through f2c to obtain C code, and then modified to improve readability. CLAPACK's goal is to provide LAPACK for someone who does not have access to a Fortran compiler.

ATLAS: The ATLAS (Automatically Tuned Linear Algebra Software) project is an ongoing research effort focusing on applying empirical techniques in order to provide portable performance. At present, it provides C and Fortran77 interfaces to a portably efficient BLAS implementation, as well as a few routines from LAPACK.

Profil: PROFIL (Programmer's Runtime Optimized Fast Interval Library) is a C++ class library supporting the most commonly needed interval and real operations in a user friendly way. PROFIL is based on BIAS (Basic Interval Arithmetic Subroutines). The developement of BIAS was guided by the ideas of BLAS, i.e. to provide an interface for basic vector and matrix operations with specific and fast implementations on various machines, the latter frequently provided by the manufacturers.

SDP solvers: SDPA (C/C++), SDPT3(Matlab), CSDP(C/C++), etc.

SDPLIB: SDPLIB is a collection of semidefinite programming test problems. All problems are stored in the SDPA sparse format.