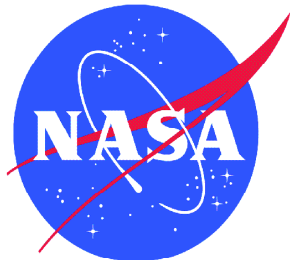


# The **StreamFEM** Application

**Tim Barth**

NASA Ames Research Center  
Moffett Field, California 94035-1000 USA



## StreamFEM

- A Brook implementation of the Discontinuous Galerkin (DG) Finite Element Method (FEM) on 2D triangulated domains
- User can change the arithmetic intensity by over 100x
  - Choice of PDE equation set
    - \* Scalar advection (1 PDE)
    - \* Euler equations (4 PDEs)
    - \* Magnetohydrodynamics (MHD) equations (6 PDEs)
  - Choice of piecewise polynomial function space
    - \* constant elements (1 dof)
    - \* linear elements (3 dofs)
    - \* quadratic elements (10 dofs)
    - \* cubic elements (15 dofs)

## StreamFEM-Examples

- Test case in Brook code is a **blast wave computation**.
- Small amount of additional code needed for high order **shock capturing**

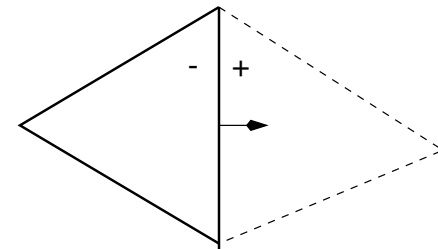
## StreamFEM: $u_t + \operatorname{div}(\vec{f}) = 0$

### Discontinuous Galerkin Finite Element Method

Find  $u \in V_h$  such that  $\forall w \in V_h$

$$\sum_{\text{elements}} \left( \int_K w u_t d\vec{x} - \int_K \vec{f}(u) \cdot \nabla w d\vec{x} + \int_{\partial K} w_- h(n; u_-, u_+) \right) = 0$$

- use  $L_2$ -orthogonal functions so that  $\int_K w u_t d\vec{x}$  yields a diagonal mass matrix suitable for explicit time stepping.
- $h(n; u_-, u_+)$  is a numerical flux such that  $h(n; u_-, u_+) = -h(-n; u_+, u_-)$



## StreamFEM–Preliminary Remarks

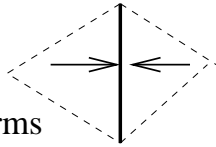
As a starting point, keep the implementation simple

- Explicit time stepping
- First order conservation laws
- No adaptive meshing
- Preprocess the mesh data structure as much as possible
  - Cuthill-McKee ordering of mesh elements and edges to improve Gather/Scatter memory performance

# StreamFEM

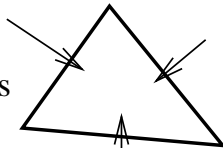
## Parsimonious wrt Arithmetic Ops

1) Gather 2 states



2) Compute trace flux terms

3) Store fluxes to memory



4) Gather 3 trace flux terms

5) Compute interior term and update

6) Store updated state

$n = \#pdes$ ,  $p = \text{polynomial order}$

1)  $n (p+1) (p+2)$  words/edge

3)  $n (p+1) (p+2)$  words/edge

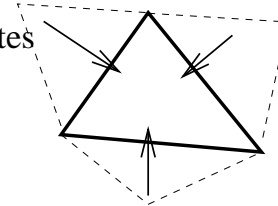
4)  $3/2 n (p+1) (p+2)$  words/edge  
 $+ n/2 (p+1) (p+2)$  words/cell

5)  $n/2 (p+1)(p+2)$  words/cell

Total:  $(10/3 n (p+1) (p+2) + \text{const})$  words/cell

## Parsimonious wrt Memory Access

1) Gather (3+1) states



2) Compute trace fluxes, interior term, and update

3) Store updated state

1)  $4/2 n (p+1) (p+2)$  words/cell

3)  $n/2 (p+1) (p+2)$  words/cell

Total:  $(5/3 n (p+1) (p+2) + \text{const})$  words /cell

## StreamFEM–Implementation

```
////////////////////////////////////  
// Gather states and compute flux for interior edges  
//  
kernel void GatherFluxInterior(Master* master,  
    FemEdgeStream edge_in_stream,  
    FemDataStream data_left_in_stream,  
    FemDataStream data_right_in_stream,  
    out FemFluxStream flux_out_stream) {  
  
    .  
    .  
    .  
}
```

## StreamFEM-Implementation

```
#include "GlobalDefines.h"

////////////////////////////////////

// Master element. Allocate and compute only once!
struct MasterElement {
    // quadrature points
    double qploc1d[QUADRATURE_POINTS_1D][2];
    double qpwt1d[QUADRATURE_POINTS_1D];
    double qploc2d[QUADRATURE_POINTS_2D][3];
    double qpwt2d[QUADRATURE_POINTS_2D];
    // Trace and element shape functions
    double shape1d[QUADRATURE_POINTS_1D][(POLY_ORDER+1)*(POLY_ORDER+2)/2][3];
    double shape2d[QUADRATURE_POINTS_2D][(POLY_ORDER+1)*(POLY_ORDER+2)/2];
    double gradShape2d[QUADRATURE_POINTS_2D][(POLY_ORDER+1)*(POLY_ORDER+2)/2][2];
};
```

## StreamFEM–Implementation

```
////////////////////////////////////  
// Gather and compute cell contribution  
kernel void GatherCellKernel(Master* master,  
                             FemCellStream cell_in_stream,  
                             FemDataStream data_in_stream,  
                             FemDataStream data_celledge_in_stream0,  
                             FemDataStream data_celledge_in_stream1,  
                             FemDataStream data_celledge_in_stream2,  
                             out FemDataStream residual_out_stream) {  
  
    .  
    .  
    .  
}
```

## StreamFEM–Implementation

```
////////////////////////////////////  
// Advance cell dofs in time  
kernel void AdvanceCellKernel(const double dt, reduce double* sum,  
    FemCellStream cell_in_stream,  
    FemDataStream residual_in_stream,  
    FemDataStream soln_in_stream,  
    out FemDataStream soln_out_stream)  
{  
    .  
    .  
    .  
}
```

## Brook Wish List (so far)

- Stream declarations in `#include` files ?
- Arrays of streams does not seem to work  

```
kernel void MyKernel(DataStream data_stream[3])
```
- Inlined functions in kernels?
- *streamLoad* and *streamStore* with constant stride argument